

**UNIVERSIDADE FEDERAL DE SERGIPE
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA
COMPUTAÇÃO**

ÍTALO PEREIRA TELES

**BPM4SERVICES: *FRAMEWORK* DIRIGIDO A MODELOS PARA
AUTOMAÇÃO DE PROCESSOS DE NEGÓCIO**

**SÃO CRISTÓVÃO/SE
2017**

**UNIVERSIDADE FEDERAL DE SERGIPE
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA
COMPUTAÇÃO**

ÍTALO PEREIRA TELES

**BPM4SERVICES: *FRAMEWORK* DIRIGIDO A MODELOS PARA
AUTOMAÇÃO DE PROCESSOS DE NEGÓCIO**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação (PROCC) da Universidade Federal de Sergipe (UFS) como parte de requisitos para obtenção do título de Mestre em Ciência da Computação.

Orientadora: Profa. Dra. Adicinéia Aparecida de Oliveira

**SÃO CRISTÓVÃO/SE
2017**

ÍTALO PEREIRA TELES

**BPM4SERVICES: *FRAMEWORK* DIRIGIDO A MODELOS PARA
AUTOMAÇÃO DE PROCESSOS DE NEGÓCIO**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação (PROCC) da Universidade Federal de Sergipe (UFS) como parte de requisitos para obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

Profª. Dra. Adicinéia Aparecida Oliveira, Orientadora
Universidade Federal de Sergipe (UFS)

Prof. Dr. Douglas Dyllon Jerônimo de Macedo, Membro Interno
Universidade Federal de Santa Catarina (UFSC)

Prof. Dr. Roquemar de Lima Baldam, Membro Externo
Instituto Federal do Espírito Santo (IFES)

**BPM4SERVICES: *FRAMEWORK* DIRIGIDO A MODELOS PARA
AUTOMAÇÃO DE PROCESSOS DE NEGÓCIO**

Este exemplar corresponde à dissertação de
Mestrado de Ítalo Pereira Teles aprovada pela
Banca Examinadora

São Cristóvão, 23 de fevereiro de 2017.

Profa. Dra. Adicinéia Aparecida Oliveira
Orientadora

Prof. Dr. Douglas Dyllon Jerônimo de Macedo
Membro Interno

Prof. Dr. Roquemar de Lima Baldam
Membro Externo

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL
UNIVERSIDADE FEDERAL DE SERGIPE

Teles, Ítalo Pereira

T269b BPM4services: framework dirigido a modelos
para automação de processos de negócios / Ítalo Pereira
Teles; orientador Adicinéia Aparecida de Oliveira - São
Cristóvão, 2017.

111 f.: il.

Dissertação (Mestrado em Ciência da
Computação) - Universidade Federal de Sergipe, 2017.

1. Arquitetura de software. 2. Framework (Arquivo de
computador). 3. Arquitetura orientada a serviços (Computador).
4. Negócios - Processamento de dados. I. Oliveira, Adicinéia
Aparecida de, orient. II. Título.

CDU 004.415.22

RESUMO

Organizações estão cada vez mais conscientes da importância da definição e gerenciamento de seus processos, dessa forma, a Gestão de Processos de Negócio (*Business Process Management* - BPM) tem evoluído ao longo dos últimos anos. Conceitos como Arquiteturas Orientadas a Serviços (*Service Oriented Architecture* - SOA) auxiliam na aproximação entre negócio e tecnologia, facilitando a comunicação e entendimento das reais necessidades do negócio. Adicionalmente, o paradigma *Model Driven Engineering* (MDE) promove a criação de softwares baseados em modelos, o qual visa aumentar a produtividade no desenvolvimento de soluções independentes de plataformas e com um menor custo de implementação. Nesse contexto, este trabalho apresenta o BPM4Services, um *framework* dirigido a modelos para automação de processos de negócio baseado em uma Arquitetura Orientada a Serviços. O *framework* tem como objetivo direcionar o desenvolvimento de soluções orientadas a processos de forma eficiente e padronizada, integrando os conceitos de BPM, SOA e MDE. Uma análise do BPM4Services, através de um caso exemplo, foi realizada no processo de Triagem Neonatal no Hospital Universitário da Universidade Federal de Sergipe, demonstrando a utilização e viabilidade do *framework*.

Palavras-chaves: Gerenciamento de Processos de Negócio, Arquiteturas Orientadas a Serviços, Abordagens Dirigidas a Modelos.

ABSTRACT

Organizations are increasingly aware of the importance of defining and managing their processes. That is why the Business Process Management (BPM) has evolved over the last few years. Concepts such as Service Oriented Architectures (SOA) help in the approximation between business and technology make easier communicating and understanding the real business needs. In addition, the Model Driven Engineering (MDE) paradigm promotes model-based software creation, which aims to increase developing platform-independent solutions productivity with lower implementation costs. In this context, this work presents BPM4Services, a model driven framework for the processes automation based on a Service Oriented Architecture. The framework intends to guide the development of solutions oriented to efficient and standardized processes, integrating concepts of BPM, SOA and MDE. An analysis of the BPM4Services, through an example, was performed in a Neonatal Screening process at the Hospital Universitário at Universidade Federal de Sergipe, demonstrating the use and viability of the framework.

Keywords: *Business Process Management, Service Oriented Architecture, Model-Driven Approaches.*

LISTA DE FIGURAS

Figura 1: Visão sistemática dos processos.	24
Figura 2: Áreas de conhecimento.	25
Figura 3: Classes do mecanismo de extensão BPMN 2.0.	31
Figura 4: Conjunto de ferramentas e tecnologias aplicadas ao BPM.	34
Figura 5: Modelo de alinhamento entre negócio e TI em uma arquitetura SOA.	35
Figura 6: Processo de Orquestração de serviços.	37
Figura 7: Coreografia de serviços.	38
Figura 8: Exemplo de código BPEL.	40
Figura 9: Relação MDE, MDD, MDA.	42
Figura 10: Modelo de domínio de uma biblioteca e seu metamodelo usado.	43
Figura 11: Principais tópicos de pesquisa em MDE.	44
Figura 12: Etapas da arquitetura MDA.	47
Figura 13: Transformação de modelos.	48
Figura 14: Regra ATL.	48
Figura 15: Protocolo da revisão sistemática.	51
Figura 16: Estrutura do BPM4Services.	58
Figura 17: Alternativas CDME.	65
Figura 18: BPMN+X.	66
Figura 19: Exemplo de um modelo BPMN+X.	66
Figura 20: Fases da metodologia para criação da extensão.	68
Figura 21: Modelo CDME para extensão B4S.ex.	71
Figura 22: Modelo BPMN+X.	78
Figura 23: Componentes adicionados na extensão.	80
Figura 24: Estrutura ATL.	81
Figura 25: Mapeamentos entre os metamodelos.	82
Figura 26: Regras de Mapeamento para Diagrama de Arquitetura.	83
Figura 27: Regras de Mapeamento para Diagrama de Participantes.	84
Figura 28: Regras de Mapeamento para Diagrama de Interfaces.	85
Figura 29: Regras de Mapeamento para Diagrama de Mensagens.	86
Figura 30: Cartão para cadastro e realização de exames da Triagem Neonatal.	90
Figura 31: Processo de Triagem Neonatal.	92

Figura 32: Diagrama de Arquitetura gerado.....	95
Figura 33: Diagrama de Participantes gerado.	96
Figura 34: Diagrama de Interface gerado.	97
Figura 35: Diagrama de Mensagens gerado.	98

LISTA DE QUADROS

Quadro 1: Principais conceitos da filosofia TQM.	20
Quadro 2: Comparativo TQM x BPR.	21
Quadro 3: Principais motivadores para implementação de BPM.	24
Quadro 4: Níveis de modelagem.	27
Quadro 5: Grupos de elementos BPMN.	28
Quadro 6: Principais componentes do padrão BPMN.	29
Quadro 7: Características da Arquitetura Orientada a Serviços.	36
Quadro 8: Resultados do processo de seleção.	52
Quadro 9: Comparativo dos artigos selecionados.	54
Quadro 10: Comparativo entre as propostas de extensões BPMN.	63
Quadro 11: Checagem de Equivalência dos novos componentes para o CDME.	68
Quadro 12: Regras para representar as propriedades das classes do CDME no modelo BPMN+X.	73
Quadro 13: Regras aplicadas às propriedades do CDME.	75
Quadro 14: Regras para representar os relacionamentos de generalizações do CDME no modelo BPMN+X.	76
Quadro 15: Regras aplicadas aos relacionamentos de generalização do CDME.	77
Quadro 16: Propriedades das Operações.	93
Quadro 17: Atributos das Entidades.	94

LISTA DE SIGLAS E ABREVIATURAS

ATL	<i>ATL Transformation Language</i>
BAM	<i>Business Activity Monitoring</i>
BPM	<i>Business Process Management</i>
BPM CBOK	<i>Business Process Management Common Body of Knowledge</i>
BPMN	<i>Business Process Modeling Notation</i>
BPEL	<i>Business Process Execution Language</i>
BPMS	<i>Business Process Management Systems</i>
BPR	<i>Business Process Reengineering</i>
CDME	<i>Conceptual Domain Model of the Extension</i>
CIM	<i>Computational Independent Model</i>
CORBA	<i>Common Object Request Broker Architecture</i>
DSML	<i>Domain-Specific Modeling Language</i>
EAI	<i>Enterprise Application Integration</i>
EPC	<i>Event-Driven Process Chain</i>
IDEF	<i>Integration Definition</i>
LWA	<i>Legacy Web Applications</i>
MDA	<i>Model Driven Architecture</i>
MDD	<i>Model Driven Development</i>
MDE	<i>Model Driven Engineering</i>
OASIS	<i>Organization for the Advancement of Structured Information Standards</i>
OMG	<i>Object Management Group</i>
PIM	<i>Platform Independent Model</i>
PSM	<i>Platform Specific Model</i>
QVT	<i>Query View Transformation</i>
RMDA	Regras de Mapeamento para o Diagrama de Arquitetura
RMDI	Regras de Mapeamento para o Diagrama de Interface
RMDM	Regras de Mapeamento para o Diagrama de Mensagens
RMDP	Regras de Mapeamento para o Diagrama de Participantes
RPC	<i>Remote Procedure Call</i>
RSL	Revisão Sistemática de Literatura
SCA	<i>Service Component Architecture</i>

SLA	<i>Service Level Agreement</i>
SOA	<i>Service Oriented Architecture</i>
TI	Tecnologia da Informação
TQM	<i>Total Quality Management</i>
UML	<i>Unified Modeling Language</i>
WSDL	<i>Web Service Description Language</i>
WSFL	<i>Web Service Flow Language</i>

SUMÁRIO

1 INTRODUÇÃO	14
1.1 PROBLEMÁTICA E MOTIVAÇÃO	15
1.2 HIPÓTESE	16
1.3 OBJETIVOS.....	16
1.3.1 OBJETIVO GERAL	16
1.3.2 OBJETIVOS ESPECÍFICOS	16
1.4 METODOLOGIA DE PESQUISA	17
1.5 ORGANIZAÇÃO DO DOCUMENTO.....	17
 2 GESTÃO DE PROCESSOS	 19
2.1 PRIMÓRDIOS DA GESTÃO DE PROCESSOS.....	19
2.1.1 <i>TOTAL QUALITY MANAGEMENT (TQM)</i>	19
2.1.2 <i>BUSINESS PROCESS REENGINEERING (BPR)</i>	20
2.1.3 COMPARATIVO ENTRE TQM E BPR	21
2.2 BUSINESS PROCESS MANAGEMENT (BPM).....	22
2.3 ÁREAS DE CONHECIMENTO BPM	25
2.4 BUSINESS PROCESS MODELING NOTATION (BPMN).....	27
2.5 MECANISMO DE EXTENSÃO BPMN 2.0	30
2.6 AUTOMAÇÃO DE PROCESSOS DE NEGÓCIO.....	32
2.6.1 <i>BUSINESS PROCESS MANAGEMENT SYSTEMS</i>	32
2.6.2 <i>SERVICE ORIENTED ARCHITECTURE (SOA)</i>	34
2.6.2.1 Características da Arquitetura Orientada a Serviços	35
2.6.2.2 Composição de Serviços.....	37
2.6.3 <i>BUSINESS PROCESS EXECUTION LANGUAGE (BPEL)</i>	39
2.7 CONSIDERAÇÕES FINAIS DO CAPÍTULO	40
 3 ABORDAGENS DIRIGIDAS A MODELOS.....	 42
3.1 VISÃO GERAL DAS ABORDAGENS DIRIGIDAS A MODELOS	42
3.2 MODEL DRIVEN ENGINEERING (MDE).....	43
3.3 MODEL DRIVEN DEVELOPMENT (MDD)	45
3.4 MODEL DRIVEN ARCHITECTURE (MDA).....	46
3.5 TRANSFORMAÇÕES DE MODELOS.....	47

3.6 CONSIDERAÇÕES FINAIS DO CAPÍTULO	49
4 BPM E SOA NA PERSPECTIVA DO DESENVOLVIMENTO ORIENTADO A MODELOS	50
4.1 PROTOCOLO DA REVISÃO SISTEMÁTICA.....	50
4.2 RESULTADOS DA REVISÃO SISTEMÁTICA	52
4.3 RESUMOS DOS TRABALHOS.....	53
4.4 ANÁLISE DOS RESULTADOS	54
4.5 CONSIDERAÇÕES FINAIS DO CAPÍTULO	56
5 FRAMEWORK BPM4SERVICES.....	57
5.1 ESTRUTURA DO BPM4SERVICES	57
5.2 EXTENSÕES BPMN	60
5.2.1 METODOLOGIA.....	63
5.2.2 ANÁLISE DO DOMÍNIO E CHECAGEM DE EQUIVALÊNCIA	68
5.2.3 MODELO CDME	70
5.2.4 MODELO BPMN+X	72
5.2.5 NOVAS PROPRIEDADES E ELEMENTOS GRÁFICOS DA EXTENSÃO.....	79
5.3 TRANSFORMAÇÃO: B4S.EX PARA SOAML	81
5.3.1 MAPEAMENTO ENTRE OS METAMODELOS DA EXTENSÃO PROPOSTA E SOAML	82
5.4 CONSIDERAÇÕES FINAIS DO CAPÍTULO	86
6 CASO EXEMPLO: TRIAGEM NEONATAL	88
6.1 O HOSPITAL UNIVERSITÁRIO E O PROGRAMA DE TRIAGEM NEONATAL (PNTN).....	88
6.2 O PROCESSO DE TRIAGEM NEONATAL NO HU.....	89
6.3 MODELO B4S.EX DA TRIAGEM NEONATAL	91
6.4 APLICAÇÃO DAS REGRAS E MODELOS SOAML GERADOS	94
6.4.1 DIAGRAMA DE ARQUITETURA	95
6.4.2 DIAGRAMA DE PARTICIPANTES	95
6.4.3 DIAGRAMA DE INTERFACES.....	96
6.4.4 DIAGRAMA DE MENSAGENS	98
6.5 ANÁLISE DA APLICAÇÃO DO BPM4SERVICES NO CASO EXEMPLO	99
6.6 CONSIDERAÇÕES FINAIS DO CAPÍTULO	100
7 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS	101
7.1 PRINCIPAIS CONTRIBUIÇÕES.....	102

7.2 LIMITAÇÕES DA PESQUISA	103
7.3 TRABALHOS FUTUROS	104
REFERÊNCIAS.....	105

CAPÍTULO 1

INTRODUÇÃO

Ao longo do século XX, durante várias décadas, como herança da Revolução Industrial Inglesa, as empresas mantiveram-se com uma estrutura fechada, organização rígida, pesada e hierárquica (GONÇALVES e DREYFUSS, 1995). Por volta das últimas décadas do século XX, as empresas começaram a abandonar a visão tradicional, funcional e hierarquizada em busca de uma estrutura horizontal e organizada por processos. Essa evolução de conceitos buscava aumentar a eficiência na prestação de serviços, maior flexibilidade frente às mudanças, melhor integração entre equipes, garantindo uma maior capacidade de aprendizagem por parte dos envolvidos nos processos (GONÇALVES, 2000).

De acordo com Graham e LeBaron (1994), todo trabalho importante realizado nas empresas faz parte de algum processo. Se existe um produto ou serviço oferecido por uma empresa, existe um processo de negócio envolvido. Para Hammer e Champy (1994), um processo é um grupo de atividades realizadas numa sequência lógica com o objetivo de produzir um bem ou um serviço que tem valor para um grupo específico de clientes.

Nesse contexto, ao longo dos anos, percebeu-se a necessidade de utilizar tecnologias para automatizar determinados processos em uma organização, aumentando sua eficiência e promovendo melhorias em seu desempenho. Dessa forma, surge o conceito de Gerenciamento de Processos de Negócio ou *Business Process Management* (BPM). BPM é apresentado como um conceito de gestão, o qual tem como objetivo alinhar os processos de negócio aos objetivos estratégicos da organização e necessidades do cliente, com auxílio da tecnologia, a fim de aumentar vantagens competitivas (CAPOTE, 2011).

Aliado ao BPM, o padrão de Arquiteturas Orientadas a Serviços ou *Service Oriented Architecture* (SOA) tem se mostrado um importante meio para integrar tecnologia e negócio, promovendo uma maior qualidade e flexibilidade nas soluções geradas. Em outras palavras, a combinação de BPM e SOA aumentam as chances de sucesso de um projeto e tornam-se uma poderosa estratégia para empresas (SOUZA e RABELO, 2010) (KAMOUN, 2007).

Porém, alcançar os benefícios gerados pela implantação de BPM e SOA em uma organização é uma tarefa complexa e, muitas vezes, lenta. Dessa forma, faz-se necessário buscar abordagens mais eficientes e sistematizadas, com melhores mecanismos de controle e maior produtividade para desenvolver soluções. Portanto, uma alternativa pode ser a integração

da Engenharia Dirigida a Modelos (MDE) aos conceitos de BPM e SOA, a qual baseia-se na construção de modelos como artefatos principais no processo de desenvolvimento de software (MILICEV, 2009).

1.1 Problemática e Motivação

De acordo com Capote (2011), os impactos positivos de se adotar BPM já estão evidenciados e comprovados pelo mercado. O autor defende que BPM promove, dentre outros benefícios, a redução expressiva de custos, aumento do retorno sobre investimento (ROI) e auxilia tomadas de decisões através do monitoramento de processos.

Entretanto, ainda existem dificuldades para lidar com ambientes dinâmicos, os quais exigem uma constante evolução dos requisitos dos processos de negócio (SOUZA e RABELO, 2010). Kirkham *et al.* (2011) também discutem a dificuldade em orquestrar serviços utilizando BPM em ambientes dinâmicos, visto que exigem complexos requisitos de flexibilidade no processo de negócio. France e Rumpe (2007) apontam a lacuna entre o domínio do problema e o domínio da implementação de uma solução como um fator impactante para aumentar a dificuldade no desenvolvimento de softwares complexos e, consequentemente, o número de problemas gerados durante sua implementação.

Sendo assim, uma forma de abordar a complexidade no desenvolvimento de soluções em ambientes dinâmicos é integrar a Gestão de Processos, juntamente com a Arquitetura Orientada a Serviços, com a Engenharia Dirigida a Modelos, a qual descreve abordagens de desenvolvimento de software, cujos modelos são criados e sistematicamente transformados em implementações concretas (WOODSIDE, PETRIU e FAISAL, 2014). A MDE ficou conhecida na indústria e academia como uma forma de enfrentar a complexidade dos softwares modernos, sendo considerada o próximo nível de abstração no desenvolvimento de sistemas (HUTCHINSON, WHITTLE e ROUCENFIELD, 2014).

A Engenharia Dirigida a Modelos apresenta inúmeros benefícios, dentre eles: aumento da produtividade, portabilidade, facilidade de manutenção, integração e evolução dos softwares. No contexto de BPM e SOA, a geração automática de serviços a partir da definição dos modelos de processos de negócio possibilita o reuso do conhecimento, o qual diminui erros em tempo de projeto, permite rastrear elementos e suas relações entre diferentes modelos, diminui a necessidade de inspeção de código, dentre outros (HUTCHINSON, WHITTLE e ROUCENFIELD, 2014) (DELGADO, GUZMÁN e PIATTINI, 2010).

No entanto, os resultados obtidos a partir de uma revisão sistemática de literatura, detalhada no capítulo 4 deste trabalho, apontaram que a maioria dos trabalhos publicados não apresentam uma solução completa para uma abordagem dirigida a modelos utilizando conceitos de BPM e SOA. As propostas abrangem pequenas etapas ou atividades de transformações de modelos sem se preocupar com as demais fases do processo.

Nesse contexto, este trabalho propõe a criação de um *framework* dirigido a modelos para automação de processos de negócio com base em arquiteturas orientadas a serviços, com o objetivo de direcionar o desenvolvimento de soluções orientadas a processos de forma eficiente e padronizada, aumentando a velocidade de produção, bem como a qualidade da solução gerada.

1.2 Hipótese

Este trabalho levanta a hipótese de que a utilização de uma abordagem dirigida a modelos poderá facilitar e agilizar o desenvolvimento de soluções orientadas a processos, alinhando tecnologia e negócio aos objetivos estratégicos da organização.

1.3 Objetivos

Esta seção descreve os objetivos deste trabalho, expondo o objetivo geral, que guia o trabalho e os objetivos específicos, os quais auxiliam sua execução.

1.3.1 Objetivo Geral

Propor o BPM4Services, *framework* dirigido a modelos para automação de processos de negócio baseado em uma arquitetura orientada a serviços.

1.3.2 Objetivos Específicos

Para alcançar o objetivo proposto nesta pesquisa, alguns objetivos específicos foram definidos, a saber:

- Avaliar propostas de abordagens dirigidas a modelos para automação de processos de negócio, identificando lacunas e dificuldades existentes.
- Definir um *framework* dirigido a modelos com objetivo de suprir às necessidades identificadas.
- Definir uma extensão do metamodelo BPMN a fim de aumentar a precisão e qualidade das transformações entre modelos.

- Definir as regras de mapeamento executadas nas transformações entre os metamodelos utilizados no *framework*.
- Construir um caso exemplo de utilização do *framework* proposto com base em um problema real identificado no Hospital Universitário da Universidade Federal de Sergipe.

1.4 Metodologia de Pesquisa

A pesquisa realizada nesse trabalho é caracterizada por abordar uma problemática a partir de soluções diferentes das utilizadas anteriormente. Dessa forma, é possível obter resultados não alcançados em trabalhos desenvolvidos anteriormente a partir da mesma problemática (WAZLAWICK, 2008). Pode-se destacar as seguintes modalidades de pesquisa científica utilizadas nesta proposta:

- Quanto à natureza de pesquisa: Pesquisa aplicada, a qual tem a necessidade de produzir conhecimento com a finalidade de contribuir para fins práticos (BARROS e LEHFELD, 2000).
- Quanto à abordagem do problema: Pesquisa qualitativa, a qual o pesquisador analisa os dados indutivamente (SILVA e MENEZES, 2001).
- Quanto à natureza das fontes: Pesquisa bibliográfica, a qual utilizará livros, artigos publicados, *websites*, dissertações e teses relacionados ao tema deste trabalho.

Como auxílio para realização deste trabalho, também foi utilizado o método de revisão sistemática de literatura que, segundo Kitchenham (2004), é uma forma de identificar, avaliar e interpretar todas as pesquisas relevantes disponíveis sobre um determinado tópico ou fenômeno de interesse. A revisão sistemática contribuiu para o levantamento das principais referências dentro do contexto e objetivos deste trabalho.

1.5 Organização do Documento

Este documento está organizado em sete capítulos, contando com esta introdução. Os tópicos a seguir descrevem os assuntos abordados em cada um dos capítulos restantes, sendo eles:

- Capítulo 2: apresenta os principais conceitos a respeito do gerenciamento de processos de negócio e tecnologias envolvidas em sua automação.

- Capítulo 3: apresenta os conceitos da Engenharia Dirigida a Modelos e o padrão de arquitetura proposto pela *Object Management Group* (OMG), o *Model Driven Architecture* (MDA).
- Capítulo 4: apresenta uma revisão sistemática de literatura levantando as principais abordagens dirigidas a modelos que utilizam conceitos de SOA e BPM.
- Capítulo 5: descreve o *framework* BPM4Services, apresentando suas principais características e camadas, bem como apresenta a extensão da notação BPMN criada, a B4S.ex.
- Capítulo 6: apresenta um caso exemplo do Processo de Triagem Neonatal do HU-UFS, o qual ilustra a transformação dos modelos CIM (B4S.ex) para o PIM (SoaML), que corresponde a uma das etapas do BPM4Services.
- Capítulo 7: apresenta as considerações finais do trabalho, identificando as principais contribuições, as dificuldades e limitações encontradas, bem como os trabalhos que poderão ser desenvolvidos.

CAPÍTULO 2

GESTÃO DE PROCESSOS

Neste capítulo serão abordados conceitos que antecederam e contribuíram ao longo dos anos para o surgimento do *Business Process Management* (BPM). Também será apresentada a definição de BPM e suas áreas de conhecimento, bem como as tecnologias que dão suporte ao ciclo de vida de um processo e sua automação.

2.1 Primórdios da Gestão de Processos

De acordo com Smith e Fingar (2007), a evolução do conceito da Gestão de Processos ocorre em três ondas: a onda inicial foi a denominada *Total Quality Management* (TQM) ou Qualidade Total, iniciada nos anos 50 a partir de Deming e Juran; a segunda, iniciada na década de 90, desenvolvia-se a ideia da Reengenharia de Processos, sendo difundida por Davenport e Hammer; finalmente, a terceira onda surge para abordar e tratar as deficiências dos modelos de gestão anteriores, iniciou-se a era do *Business Process Management* ou Gerenciamento de Processos de Negócio. As próximas seções apresentam maiores detalhes acerca das características de cada modelo de gestão.

2.1.1 *Total Quality Management* (TQM)

Para Mansir e Schacht (1989), *Total Quality Management* é um meio para aprimorar a eficiência e desempenho, bem como alinhar e direcionar os esforços individuais em toda organização. Dessa forma, TQM apresenta um conjunto de princípios e práticas, os quais permitem desenvolver uma organização criada e sustentada por uma cultura comprometida com a melhoria contínua de suas atividades.

TQM tenta incorporar que qualidade atinge cada aspecto da organização, abordando desde a perspectiva técnica até o envolvimento e compreensão das pessoas com a visão de qualidade. Uma de suas características é identificar a causa raiz dos problemas e corrigi-los na fonte, ao invés de resolvê-los apenas após o produto ter sido construído (SANDERS e REID, 2012). O Quadro 1 são apresentados os principais conceitos que compõem a filosofia TQM, segundo Sanders e Reid (2012).

Quadro 1: Principais conceitos da filosofia TQM.

Conceito	Principal ideia
Foco no cliente	Objetivo é identificar e conhecer as necessidades dos clientes.
Melhoria contínua	A filosofia de nunca parar de melhorar.
Capacitação dos funcionários	Funcionários devem procurar, identificar e corrigir problemas de qualidade.
Uso de ferramentas de qualidade	Formação continuada dos funcionários no uso de ferramentas de qualidade.
Projeto dos produtos	Produtos precisam ser projetados para atender às expectativas dos clientes.
Gerenciamento de processos	Qualidade deve ser construída dentro de um processo; fontes de problemas de qualidade devem ser identificados e corrigidos.
Gestão da qualidade do fornecedor	Conceitos de qualidade devem estender-se aos fornecedores.

Fonte: Traduzido de Sanders e Reid (2012).

2.1.2 *Business Process Reengineering (BPR)*

A rigidez ocasionada pelas estruturas de negócios mais tradicionais sufocava a inovação e criatividade dentro das organizações. O foco nas atividades ao invés dos resultados dos produtos ou serviços gerados e o excesso de burocracia nas tarefas proporcionavam atrasos no desenvolvimento das empresas (VIDOTTI *et al.*, 1998).

Por volta dos anos 90, surge uma nova filosofia de gerenciamento focada no aumento da competitividade de mercado, o *Business Process Reengineering* (BPR) ou Reengenharia de Processos de Negócio. Davenport e Short (1990), em um dos primeiros artigos publicados sobre BPR, defendiam o uso da Tecnologia da Informação (TI) e redesenho de processos como uma forma de transformar e melhorar os processos de negócio de uma organização. Para tanto, os autores propuseram uma metodologia de 5 etapas para realizar o redesenho dos processos. A metodologia iniciava com a etapa responsável por projetar a visão do negócio e seus objetivos. Durante a segunda etapa, era escolhido o processo que deveria ser repensado, aquele que representasse o maior impacto para organização. Em seguida, era necessário medir e entender os problemas existentes no processo e planejar as melhorias futuras. A quarta etapa identificava como a TI poderia ser usada para prover melhores formas de remodelar o processo. Finalmente,

compondo a quinta etapa da metodologia, um protótipo era criado para simular e testar o novo processo antes de realizar sua completa implementação.

Michael Hammer (1990), com uma visão um pouco mais radical da reengenharia, alegava que os esforços da racionalização e automação do passado não aumentavam significativamente a eficiência dos processos. O cenário estava diferente da produção em massa dos anos 80, agora, por outro lado, a competição era mais intensa, os clientes estavam mais exigentes, ou seja, mudanças constantes eram necessárias para se manter no mercado. Dessa forma, Hammer (1990) acreditava que os processos deveriam ser reestruturados radicalmente para obter o real benefício da BPR, aplicando a tecnologia a seu favor e atendendo às necessidades dos clientes.

Contudo, os processos deveriam ser repensados de maneira ampla, ao invés de realizar ajustes para resolver problemas através de pequenas melhorias incrementais. De maneira geral, criticava-se a forma de organização funcional e hierarquizada das empresas, caracterizando-a como rígida e de difícil adaptabilidade, enquanto que defendia-se a orientação a processos, apresentando-a como superior em coordenação e medição de performance alinhada aos objetivos do processo (CHANG, 2006).

2.1.3 Comparativo entre TQM e BPR

Embora ambas abordagens reconheçam a importância da gestão de processos e atenção às necessidades dos clientes, elas diferem em alguns pontos. Por exemplo, no que se refere ao foco e magnitude da mudança proposta, enquanto TQM acredita na reformulação de pequenos processos, através de melhorias contínuas e incrementais de sua qualidade, BPR recomenda uma remodelagem dramática e radical em como toda a organização executa suas atividades e gere seus negócios (SHEN e LAU, 1995) (LUSK, PALEY e SPANYI, 2005) (CHANG, 2006). O Quadro 2 apresenta um comparativo entre TQM e BPR quanto ao seu foco, escopo, nível de mudança, grau de risco e alguns métodos e ferramentas que suportam cada abordagem.

Quadro 2: Comparativo TQM x BPR.

Característica	TQM	BPR
Foco	Gerenciamento da qualidade, eficiência da tarefa.	Inovação e reformulação dos principais processos da organização.
Escopo	Pequenos processos ou atividades.	Processos interfuncionais, os quais passavam por diferentes áreas da organização.

Característica	TQM	BPR
Nível de mudança	Parte do princípio que o propósito fundamental do processo ou atividade ficará intacto.	Assume que o processo inteiro deverá ser reformulado a partir do zero.
Risco	Moderado.	Alto.
Principal objetivo	Melhoria da qualidade.	Redução de custos.
Ferramentas/Métodos	Controles estatísticos dos processos, métodos de melhorias contínuas das atividades.	Métodos de reengenharia, processos de reformulação.

Fontes: Shen e Lau (1995); Lusk, Paley e Spanyi (2005); Chang (2006).

Por volta da metade dos anos 90, uma característica dentro de todas as diferentes abordagens orientadas a processos destacou-se: o gerenciamento de processos. Davenport e Stoddard (1994) previram, como futuro da Reengenharia, a combinação da BPR com outras iniciativas orientadas a processos a fim de criar um modelo integrado de gerenciamento de processos. Hammer (2002) apresentou uma visão de como TQM, BPR e suas variações poderiam trabalhar em conjunto para conseguir gerenciar processos. Dessa forma, estes e outros trabalhos foram o início de um novo conceito, o qual englobava características de diversas abordagens orientadas a processo e que hoje ficou conhecido como *Business Process Management* (BPM).

2.2 *Business Process Management* (BPM)

De acordo com Gonçalves (2000), empresas são coleções de processos, não existe um produto ou serviço oferecido por uma empresa sem um processo associado. Dessa forma, para definir e entender o conceito de *Business Process Management* (BPM) é fundamental entender e definir o significado de “processo” no âmbito dos negócios. Davenport (1993), define processo como uma estrutura composta por um conjunto de atividades mensuráveis, as quais são projetadas para produzirem um resultado único para um consumidor ou mercado específico. Para Hammer e Champy (1994), processo é uma coleção de atividades, cujo objetivo é criar um produto ou serviço que tenha valor para o cliente. O *Business Process Management Common Body of Knowledge* (ABPMP BPM CBOK v3.0, 2013), define processo como “(...) uma

agregação de atividades e comportamentos executados por humanos ou máquinas para alcançar um ou mais resultados.”.

Em resumo, todas as definições apresentadas destacam que o propósito de qualquer processo é realizar uma transformação dos recursos que entram (materiais, formas de energia, informações, dentre outros) em resultados (produtos ou serviços) com valor agregado ao cliente. A Figura 1 apresenta um modelo genérico de funcionamento de um processo em uma organização. Pode ser observado que as saídas de um processo podem ser informações que realimentarão o sistema organizacional, servindo como um sistema de *feedback*, proporcionando melhorias no funcionamento do processo. A imagem também ilustra os diversos recursos utilizados nas transformações de entradas em saídas, bem como as influências externas à organização (influências políticas, jurídicas, do mercado, dentre outros.) (BALDAM *et al.*, 2009).

Nesse contexto, BPM é apresentado como um conceito de gestão, o qual tem como objetivo alinhar os processos de negócio aos objetivos estratégicos da organização e necessidades do cliente, com auxílio da tecnologia, a fim de aumentar vantagens competitivas (CAPOTE, 2011). O BPM CBOK V3.0 (2013, p. 40) diz que “BPM é uma disciplina gerencial que integra estratégias e objetivos de uma organização com expectativas e necessidades de clientes, por meio do foco em processos ponta a ponta”. Já Capote (2011, p. 48) define Gerenciamento de Processos de Negócio como:

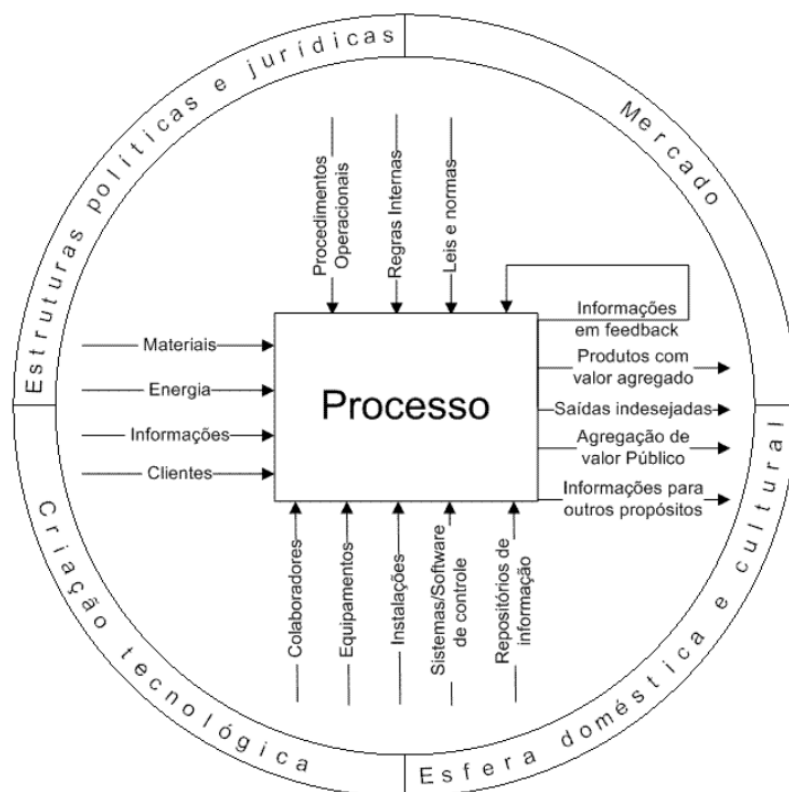
Uma abordagem disciplinar para identificar, desenhar, executar, documentar, medir, monitorar, controlar e melhorar processos de negócio, automatizados ou não, para alcançar resultados consistentes e alinhados com os objetivos estratégicos da organização.

Uma pesquisa realizada pelo Winter Green Research em 2014¹ levantou dados sobre os investimentos atuais e previstos em *Business Process Management*. A análise realizada indicou um gasto atual de \$3.4 bilhões com previsão de alcançar \$10 bilhões até 2020. Conforme ilustrado no Quadro 3, através de uma pesquisa realizada pela BPTrends (2014)², é possível visualizar os principais motivadores para implantação de BPM. De acordo com os dados levantados, o principal motivador identificado foi a necessidade de reduzir custos e melhorar produtividade, seguido pela melhoria da satisfação do cliente, ambos tentam aumentar a competitividade da organização no mercado.

¹ *Business Process Management (BPM) Cloud, Mobile, and Patterns: Market Shares, Strategies, and Forecasts, Worldwide, 2014 to 2020.*

² Responderam esta pesquisa 309 integrantes de algum tipo de iniciativa BPM, os resultados foram publicados por Paul Harmon e Celia Wolf em “*The State of Business Process Management* (2014) ”.

Figura 1: Visão sistemática dos processos.



Fonte: Baldam *et al.* (2009).

Com a crescente demanda e utilização de BPM, fez-se necessário organizar um corpo comum de conhecimento contendo as melhores práticas e padronizando os conhecimentos obtidos por diversos profissionais da área. Este guia foi intitulado de BPM CBOK (*Business Process Management Common Body Knowledge*) e foi organizado e publicado pela ABPMP (*Association of Business Process Management Professionals*). Em sua versão 3.0, o BPM CBOK divide a disciplina de BPM em nove áreas de conhecimento, as quais serão discutidas a seguir.

Quadro 3: Principais motivadores para implementação de BPM.

Motivadores	%
Necessidade de reduzir custos e melhorar produtividade	54%
Incrementar qualidade de produtos existentes ou criar novos produtos para aumentar competitividade	34%
Eventos únicos (Fusão ou Aquisição)	03%
Governança ou gerenciamento de riscos	13%
Melhorar satisfação do cliente para aumentar competitividade	37%
Melhorar a coordenação ou atendimento organizacional	35%

Motivadores	%
Promover melhorias no gerenciamento de recursos de TI	18%
Outros	11%

Fonte: Harmon e Wolf (2014).

2.3 Áreas de conhecimento BPM

O Guia para o Gerenciamento de Processos de Negócio Corpo Comum de Conhecimento (BPM CBOK) tem como objetivo principal apresentar uma visão geral das áreas de conhecimento (ilustradas pela Figura 2) que são reconhecidas e aceitas como boas práticas. O guia também fornece as principais atividades executadas em cada etapa do ciclo de vida do gerenciamento de processos.

Figura 2: Áreas de conhecimento.



Fonte: (ABPMP BPM CBOK v3.0, 2013).

De acordo com o (ABPMP BPM CBOK v3.0, 2013), existem nove áreas de conhecimento, sendo elas:

- Gerenciamento de Processos de Negócio: área responsável por apresentar os conceitos básicos de BPM, como os tipos de processos (primários, suporte e gerenciamento), instâncias de processos, dentre outros. Também aborda o ciclo de vida de um processo, detalhando cada fase.
- Modelagem de Processos: área que trata do conjunto de atividades envolvidas na criação de modelos e representações do processo de negócio. Aborda as diferentes habilidades e técnicas que permite compreender, comunicar e

gerenciar os componentes de um processo de negócio. Relaciona e diferencia os conceitos de diagramas, mapas e modelos. Cita as principais notações para modelagem de processos do mercado (ex.: BPMN, fluxogramas, *Unified Modeling Language* - UML, *Event-Driven Process Chain* - EPC, *Integration Definition* - IDEF, dentre outras).

- **Análise de Processos:** seu principal objetivo é gerar o entendimento das atividades envolvidas no negócio a fim de medir o sucesso dessas atividades no alcance dos objetivos e metas estratégicas da organização. Ou seja, abrange as principais técnicas (ex.: entrevistas, simulações, observações, *workshops*, dentre outras) para analisar o estado atual dos processos na organização (processo “*as is*”) e identificar restrições e rupturas que interferem no desempenho do processo.
- **Desenho de Processos:** área que tem como objetivo estruturar os novos processos (processo “*to be*”), com base nos resultados da análise do estado atual (“*as is*”), alinhando-os com os objetivos estratégicos da organização e focados no cliente.
- **Gerenciamento de Desempenho de Processos:** estuda diferentes formas de medições de processos de negócio, define como e o que medir. As perspectivas básicas de análise do desempenho são tempo, custo, capacidade e qualidade do processo. Esta área também aborda conceitos e exemplos de indicadores de desempenho, definidos como representações simplificadas de uma métrica do processo comparada a uma linha base ou referência.
- **Transformação de Processos:** área responsável por estabelecer as melhores maneiras de realizar a evolução de um processo. Focam na Gerência de Mudança, responsável por utilizar técnicas e métodos para suportar a transição de um estado atual para um estado futuro de forma sustentável com o menor impacto possível nos objetivos reais da organização.
- **Organização de Gerenciamento de Processos:** nesta área de conhecimento são discutidos pontos considerados importantes na governança de processo no que diz respeito a composição da estrutura organizacional orientada a processos. Também são descritos os objetivos, papéis e responsabilidades dos escritórios de processos.

- Gerenciamento de Processos Corporativos: área que utiliza todos os conceitos das demais áreas de conhecimento de forma integrada para prover e assegurar o alinhamento entre o portfólio e a arquitetura de processos ponta a ponta com a estratégia e recursos da organização.
- Tecnologias de Gerenciamento de Processos de Negócio: apresentação das diversas tecnologias que dão suporte a todo o ciclo de vida de um processo, desde sua análise e modelagem até sua automação e gerenciamento de desempenho. São descritas diversas tecnologias com diferentes abordagens de implementação, alguns exemplos: *Business Process Management Systems* (BPMS), *Business Activity Monitoring* (BAM), *Service Oriented Architecture* (SOA), *Enterprise Application Integration* (EAI), dentre outras.

A fim de melhorar o entendimento deste trabalho os conceitos na área de conhecimento de Modelagem de Processos serão aprofundados, principalmente no que diz respeito à notação BPMN, detalhada a seguir.

2.4 Business Process Model and Notation (BPMN)

Mantida como um padrão de modelagem de processos de negócio pela OMG (*Object Management Group*), BPMN tem como objetivo prover uma notação compreensível por todos os envolvidos em um ambiente orientado a processos, desde os analistas de negócio até os desenvolvedores responsáveis por implementar tecnologias que irão suportar a execução dos processos. Ou seja, BPMN é um padrão para modelagem, o qual utiliza técnicas e componentes bem definidos voltados especialmente para definição e documentação de processos de negócio. Um estudo apresentado por Silver (2009), apresentou três níveis de modelagem, cada uma abordando seu propósito, explicados no Quadro 4.

Quadro 4: Níveis de modelagem.

Nível de Modelagem	Descrição
Modelagem Descritiva	Conta com um alto nível de abstração, focada no mapeamento e descrição dos processos de negócio, gerando sua documentação inicial simplificada, representada com poucos componentes da notação e com boa compreensão.
Modelagem Analítica	Apresenta maiores detalhes na representação formal do processo de negócio. Neste nível, formas de tratar exceções, manipular as

Nível de Modelagem	Descrição
	principais regras do negócio e definições de análises de desempenho são representadas no modelo.
Modelagem Executável	Nível que requer a maior riqueza de detalhes em sua especificação, pois será utilizado para executar o modelo de forma sistêmica. O processo de negócio modelado deve conter a semântica completa e exata de cada componente da notação, acrescentando atributos internos para compor suas regras e, assim, ser capaz de executar o processo em ferramentas apropriadas.

Fonte: Silva (2009).

Em BPMN, um processo é descrito como um fluxo de elementos apresentados graficamente, os quais são compostos por Objetos de Sequência (*Flow Object*), Dados (*Data*), Objetos de Conexão (*Connecting Objects*), *Swimlanes* e Artefatos (*Artefacts*), com o objetivo de definir um conjunto finito de uma execução semântica. O Quadro 5 apresenta os principais grupos de elementos da notação (OMG, 2011).

Quadro 5: Grupos de elementos BPMN.



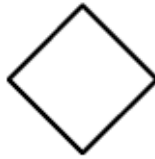


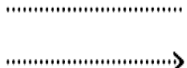
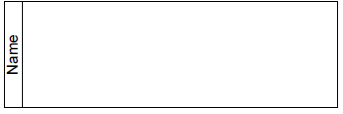
Grupo	Elementos	Descrição
Objetos de Sequência	<ul style="list-style-type: none"> • Eventos (<i>Events</i>) • Decisões (<i>Gateways</i>) • Atividades (<i>Activities</i>) 	São os principais elementos que compõem a notação. Utilizados para definir o comportamento do processo de negócio.
Dados	<ul style="list-style-type: none"> • Objetos de dados (<i>Data Objects</i>) • Entrada de dados (<i>Data Inputs</i>) • Saída de dados (<i>Data Outputs</i>) • Depósito de dados (<i>Data Store</i>) 	Representa como os dados são produzidos, requisitados ou mesmo armazenados pelas atividades.
Objetos de Conexão	<ul style="list-style-type: none"> • Fluxos de sequência (<i>Sequence Flows</i>) • Fluxos de mensagem (<i>Message Flows</i>) • Associações (<i>Associations</i>) • Associações de dados (<i>Data Associations</i>) 	Utilizados para criar o esqueleto estrutural básico de um processo de negócio, conectando seus elementos.
<i>Swimlanes</i>	<ul style="list-style-type: none"> • Piscinas (<i>Pools</i>) • Raias (<i>Lanes</i>) 	Funcionam como agrupadores dos elementos primários da notação, ilustrando diferentes atribuições ou papéis dentro do processo de negócio.
Artefatos	<ul style="list-style-type: none"> • Grupo (<i>Group</i>) • Anotação de texto (<i>Text Annotation</i>) 	Utilizados para acrescentar informações extras no modelo, porém




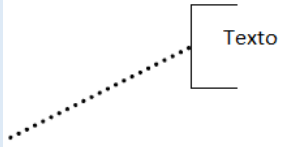
Grupo	Elementos	Descrição
		não afeta a semântica dos elementos BPMN durante a execução do processo.

Fonte: Adaptado de OMG (2011).

De acordo com Weske (2007), os elementos principais do BPMN permitem criar estruturas simples e de fácil entendimento, não sendo necessários extensos treinamentos. Contudo, para usuários mais avançados, também é possível criar modelos complexos e detalhados utilizando diferentes aspectos dos elementos disponibilizados na notação. O Quadro 6 ilustra os principais componentes disponíveis no padrão BPMN.

Quadro 6: Principais componentes do padrão BPMN.

Componente	Descrição	Notação
Eventos	Um evento é um gatilho dentro da execução do processo de negócio, o qual tem uma causa ou impacto que afeta o modelo. Existem três tipos de eventos, baseados no momento em que afetam o fluxo: Início, Intermediário e Final.	
Atividades	É o termo utilizado para representar o trabalho executado na organização. Atividades podem ser atômicas (tarefas) ou não atômicas (subprocessos).	
Decisões	Usado para controlar o direcionamento do fluxo das atividades, convergindo ou divergindo de acordo com as expressões descritas no componente.	
Fluxo de sequência	Utilizado para definir a ordem de execução do fluxo de atividades.	
Fluxo de mensagem	Utilizado para trocar mensagens entre dois participantes em um processo.	
Associação	Liga os artefatos aos elementos aos quais estão associados.	
Piscina	Representação gráfica de um participante colaborador do processo.	

Componente	Descrição	Notação
Raias	Subdivisão de uma piscina, comumente utilizado para representar diferentes categorias ou papéis envolvidos no processo.	
Objeto de dados	Representa as informações que as atividades necessitam ou produzem durante sua execução.	
Grupo	Artefato utilizado para facilitar o entendimento do modelo, agrupando elementos em uma categoria de acordo com as regras do negócio. Não afeta o fluxo de sequência das atividades.	
Anotações	Usado para acrescentar qualquer tipo de informação necessária para melhorar a documentação do processo de negócio.	

Fonte: Adaptado de OMG (2011).

Em sua versão mais recente, atualizada em 2011, o BPMN 2.0 cria um mecanismo capaz de estender os elementos originais da notação, inserindo novas propriedades ou componentes ao mesmo tempo em que mantém a conformidade com seu metamodelo. A próxima seção abordará este mecanismo de extensão do metamodelo BPMN.

2.5 Mecanismo de extensão BPMN 2.0

De acordo com Braun *et al.* (2014), utilizar as classes de extensão do BPMN com conceitos de domínio específicos tende a ter um menor custo do que desenvolver uma nova linguagem específica de domínio. O padrão BPMN disponibiliza uma abordagem para estendê-lo por adição, a qual prevê um conjunto de elementos que permitem anexar atributos e componentes de um novo domínio específico aos elementos pré-definidos da linguagem mantendo a compatibilidade com o metamodelo original (OMG, 2011).

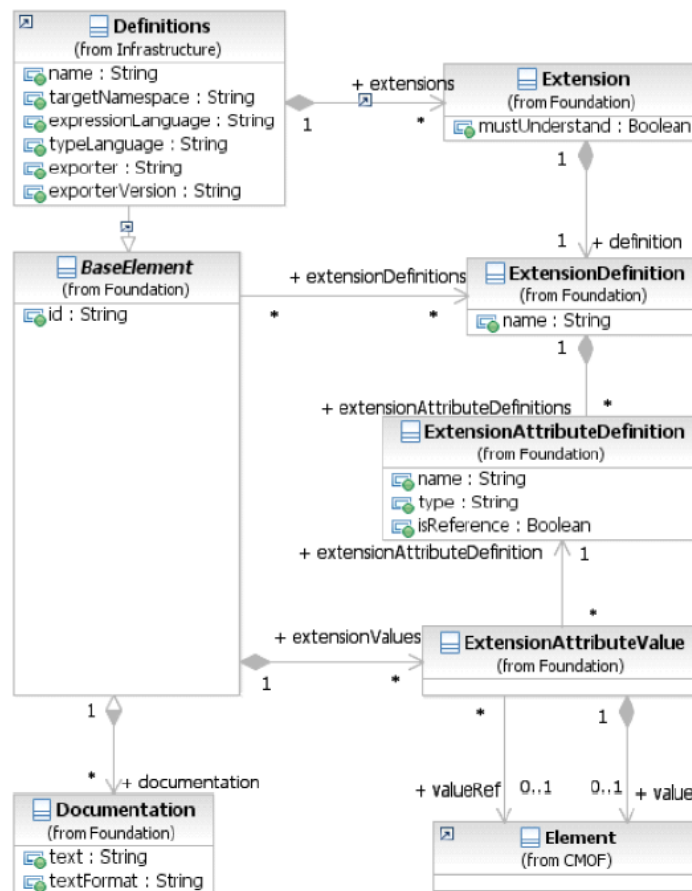
Existem basicamente quatro elementos que compõem o modelo de extensão BPMN (ilustradas na Figura 3), são elas (OMG, 2011):

- *Extension*: elemento responsável por ligar/importar uma *ExtensionDefinition* e seus atributos para a definição no modelo BPMN. O elemento *Extension* está

contido no elemento BPMN *Definition* e, por isso, permite que a *ExtensionDefinition* possa ser associada a qualquer elemento BPMN.

- *ExtensionDefinitions*: classe que define e agrupa atributos adicionais. Pode ser criada independentemente de qualquer elemento ou definição BPMN.
- *ExtensionAttributesDefinition*: definem a lista de atributos que podem ser anexados a qualquer elemento BPMN. Contém o nome e tipo dos novos atributos.
- *ExtensionAttributesValue*: contém os valores dos novos atributos.

Figura 3: Classes do mecanismo de extensão BPMN 2.0.



Fonte: (OMG, 2011).

Todo elemento BPMN que herda da classe *BaseElement* pode ser estendido com atributos adicionais. Para tanto, é necessário que o elemento seja associado a um *ExtensionDefinition*, o qual será subclasse da classe *Definition*.

Apesar da notação BPMN oferecer um modelo bem definido para criar extensões, poucos trabalhos fazem uso dele. Ao invés disso, a maioria das extensões são propostas apenas graficamente. Para Braun *et al.* (2014), isso impede, não somente uma compreensão e

comparação entre as extensões criadas, como também uma boa integração com ferramentas BPMN existentes, devido à falta de conformidade com o metamodelo.

Um dos objetivos do BPMN é criar uma ponte entre a área de negócio e a área de TI, responsável por implementar o processo em um ambiente tecnológico apropriado. Dessa forma, na próxima seção serão discutidos alguns conceitos e formas de automação de processo de negócio.

2.6 Automação de Processos de Negócio

Inúmeras organizações têm consciência que seus processos são, por vezes, demasiadamente complicados, seja pela quantidade de informação envolvida, ou mesmo pela complexidade de suas regras de negócio. Modelar processos de negócio, não se resume apenas a definir regras de comportamento, mas também identificar a forma como os processos poderão ser acionados, interagindo com diferentes ambientes e sistemas de informação, automatizando a execução das atividades do negócio. Com esta finalidade, junto a abordagem BPM, surgem tecnologias que dão suporte a automação e são capazes de integrar diferentes tipos de sistemas, são os *Business Process Management Systems*.

2.6.1 Business Process Management Systems

De acordo com Capote (2011, p. 206), BPMS pode ser definido como um “ambiente integrado de componentes de software que automatizam o ciclo de vida de processos de negócio”. Para tanto, o autor destaca as quatro principais funcionalidades encontradas em um BPMS:

- Definição dos processos: etapa responsável por apresentar a estrutura do processo de negócio através de um modelo, o qual será composto por todas as informações necessárias para que o sistema possa executar o processo.
- Controle de execução dos processos: após sua definição, o sistema será capaz de criar as instâncias daquele processo, podendo várias delas serem executadas simultaneamente. O BPMS será encarregado de delegar os recursos computacionais necessários para execução de cada atividade por cada responsável no momento adequado.
- Controle de integrações e gerenciamento: funcionalidade que apresenta aos participantes do processo suas devidas tarefas. A execução das atividades pode envolver manipulação de dados, tomada de decisões, preenchimento de

formulário ou mesmo acesso a outros sistemas ou serviços. Com o término da tarefa, o processo segue o fluxo definido de acordo com os resultados obtidos.

- Acompanhamento de execuções: permite ao usuário gerenciar as instâncias do processo em execução, podendo até mesmo realizar controle estatísticos para um melhor controle do desempenho.

Chang (2006) destaca 5 (cinco) pontos no que diz respeito à capacidade de um BPMS:

1. Aproxima o setor de TI com a área de negócio, a fim de implantar as soluções da Gestão de Processos.
2. Habilidade de integrar pessoas e sistemas que participam do processo de negócio.
3. Habilidade de simular o processo de negócio com o objetivo de projetar o melhor processo possível.
4. Capacidade de monitorar, controlar e realizar melhorias em tempo de execução do processo.
5. Capacidade de executar mudanças em um processo de negócio em tempo real sem grandes esforços na conversão.

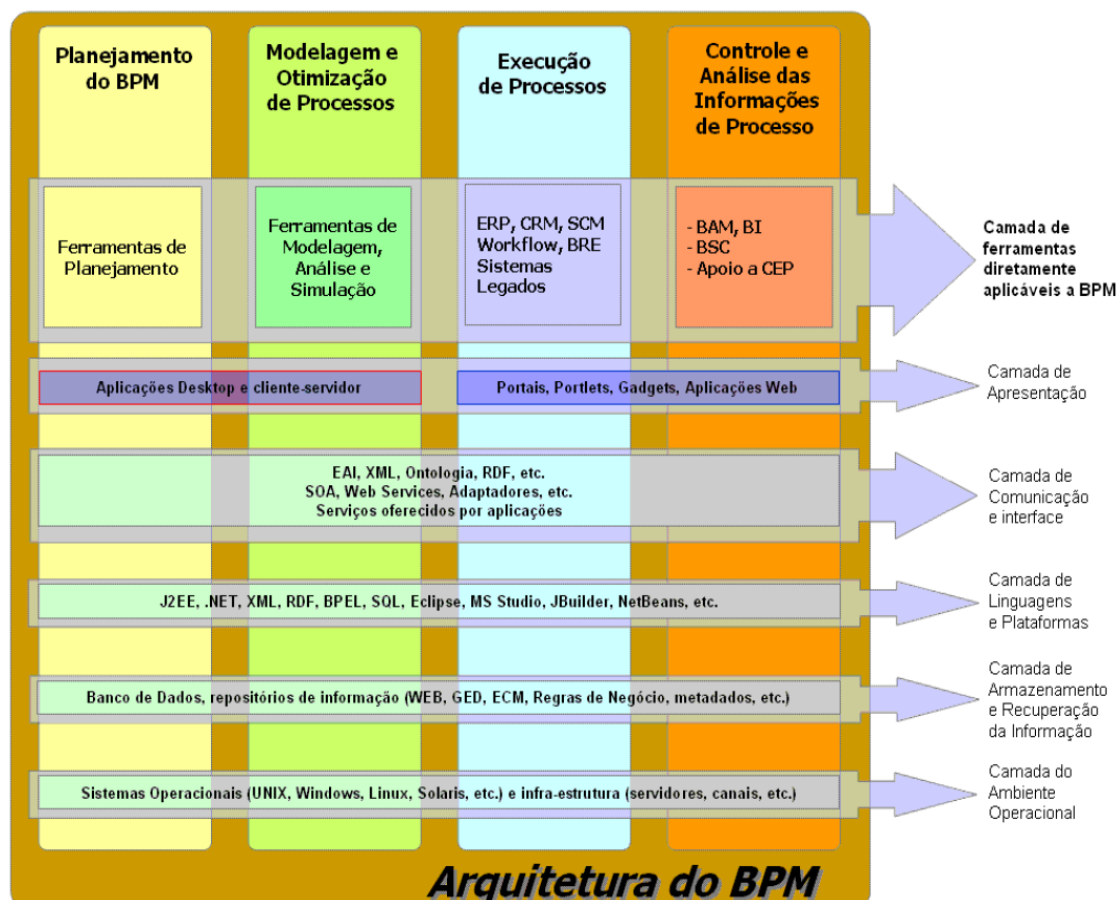
Para Baldam *et al.* (2009), uma única ferramenta não teria condições de apoiar todas as perspectivas BPM, solucionando todos seus problemas. Sendo assim, o autor afirma que antes de escolher a ferramenta a ser adotada, deve-se levantar a seguinte questão: ferramenta para fazer o que no BPM? Dessa forma, ele classificou as tecnologias aplicadas em BPM em:

- Camada de ferramentas diretamente aplicáveis a BPM: apresenta tecnologias comumente procuradas por profissionais como sendo ferramentas de BPMS.
- Camadas de infraestrutura: servem de apoio às ferramentas BPMS.

A Figura 4 apresenta diversas tecnologias, que compõem uma solução BPM. Segundo Baldam *et al.* (2009), tais tecnologias, em caso de serem comercializadas em conjunto, podem dar origem a uma ferramenta BPMS.

Uma das formas de automatizar processos de negócio é utilizar o padrão SOA para arquitetura de software, aceita pela maioria das ferramentas BPMS. A próxima seção abordará mais detalhes a respeito dessa arquitetura e como ela pode ser utilizada para contribuir com os benefícios do uso de BPM na organização.

Figura 4: Conjunto de ferramentas e tecnologias aplicadas ao BPM.



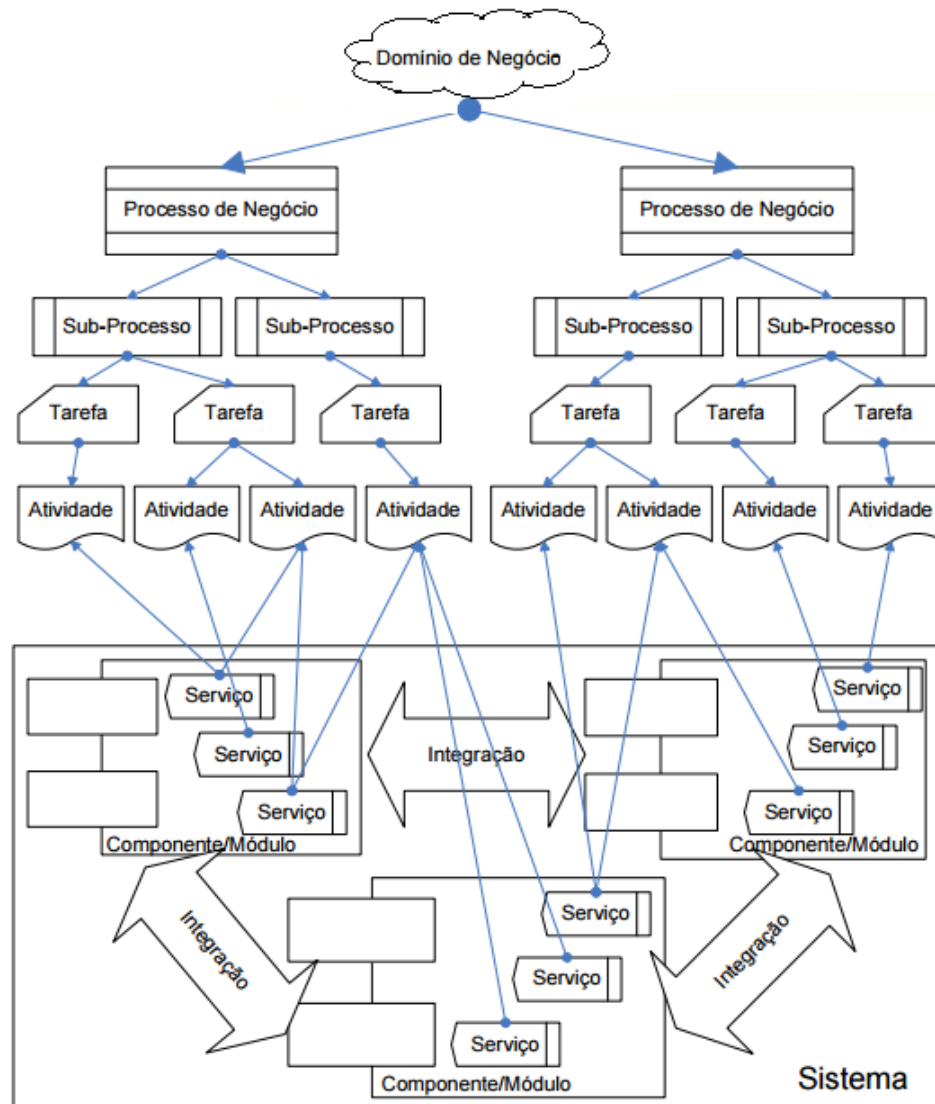
Fonte: Baldam *et al.* (2009).

2.6.2 Service Oriented Architecture (SOA)

SOA representa uma abordagem, em níveis técnico e organizacional, de como o processo de alinhamento estratégico pode ser suportado pela tecnologia da informação. Este alinhamento é realizado identificando elementos de um processo de negócio e associando-os a serviços de TI (MARZULLO, 2009). A Figura 5 ilustra, através de um modelo geral, como uma arquitetura de software baseada em serviços pode ser alinhada aos processos de negócio da organização.

Dessa forma, uma arquitetura SOA permite o desenvolvimento de uma infraestrutura para computação distribuída, através de serviços disponibilizados e consumidos dentro de uma organização e entre organizações, por meio de redes de comunicação como, por exemplo, a Internet (FUGITA, 2012).

Figura 5: Modelo de alinhamento entre negócio e TI em uma arquitetura SOA.



Fonte: Marzullo (2009).

É válido destacar que SOA não é uma tecnologia, é na verdade um modelo de arquitetura de software baseada em permitir que sistemas corporativos disponibilizem suas funcionalidades através de serviços que podem ser acessados por outros sistemas para executar suas atividades. Para Hurwitz (2009), trata-se de um modelo de arquitetura de software que utiliza conjuntos de componentes fracamente acoplados e orquestrados para implementar processos de negócio ou serviços.

2.6.2.1 Características da Arquitetura Orientada a Serviços

Arquitetura Orientada a Serviços pode fornecer a infraestrutura tecnológica necessária para que um sistema de informação possa ser definido por meio de composição de serviços, dessa forma, auxilia aplicações distribuídas de uma forma flexível e de baixo custo. A

composição de serviços é vista como um processo de negócio composto por componentes reutilizáveis e interoperáveis em serviços.

Em SOA, os recursos são disponibilizados de forma que quem acessa este serviço não tem necessidade de conhecer a sua plataforma de implementação. Esta arquitetura não está associada a nenhuma tecnologia específica e pode ser implantada de várias formas, como por exemplo *RPC (Remote Procedure Call)*, *CORBA* e *Web Services*. Com isso, serviços independentes são requisitados através de interfaces predefinidas e públicas para execução de determinadas atividades sem haver necessidade de conhecer a aplicação que solicitou o serviço e sem a aplicação conhecer como os serviços são implementados e executados.

De acordo com Miranda (2008), a composição de sistemas de informação compostos por serviços deve conter todas as características apresentadas no Quadro 7.

Quadro 7: Características da Arquitetura Orientada a Serviços.

Característica	Descrição
Fraco acoplamento	Fornecer uma maior flexibilidade ao serviço frente a mudanças da regra de negócio, pois minimizam as dependências entre as aplicações.
Reusabilidade do Serviço	As lógicas das regras de negócio devem ser estabelecidas de forma que outras aplicações possam fazer uso do serviço sem grandes esforços.
Contrato do Serviço	Os serviços devem disponibilizar de forma clara e padronizada a especificação da forma como devem ser acessados (informações de entrada e saída).
Abstração	O encapsulamento das regras de negócio através de serviços bem definidos promove um alto nível de abstração, facilitando o seu reuso.
Composição	Serviços podem ser utilizados de forma composta para gerar novos serviços, promovendo novas funcionalidades agregadas, com um nível maior de abstração e mais flexível.
Alta Granularidade	No desenvolvimento de aplicações complexas, a nível de sistemas corporativos e extensos processos de negócio, a alta granularidade traz vantagens, pois os detalhes da implementação daquele serviço são deixados sob responsabilidade da equipe de desenvolvimento.
Heterogeneidade	Serviços são construídos de forma que sejam acessados através de qualquer ambiente, independentemente de sua plataforma de desenvolvimento, tecnologias de implementação e linguagens de programação.
Ubiquidade	Os serviços devem ser disponibilizados de modo que possam ser acessados de qualquer lugar e a qualquer momento.

Fonte: Miranda (2008).

2.6.2.2 Composição de Serviços

Existe um conceito muito utilizado para aumentar o aproveitamento dos benefícios da arquitetura orientada a serviços, solucionando problemas como a distribuição e heterogeneidade de aplicações, chama-se Composição de Serviços. Esta técnica tem como objetivo combinar dois ou mais serviços a fim de que, juntos, possam oferecer funcionalidades que vão além das suas capacidades individuais. Ou seja, a composição de serviços é baseada em padrões bem definidos, podendo criar processos de negócio de alto nível, com um alto valor agregado (INAZAWA, 2009).

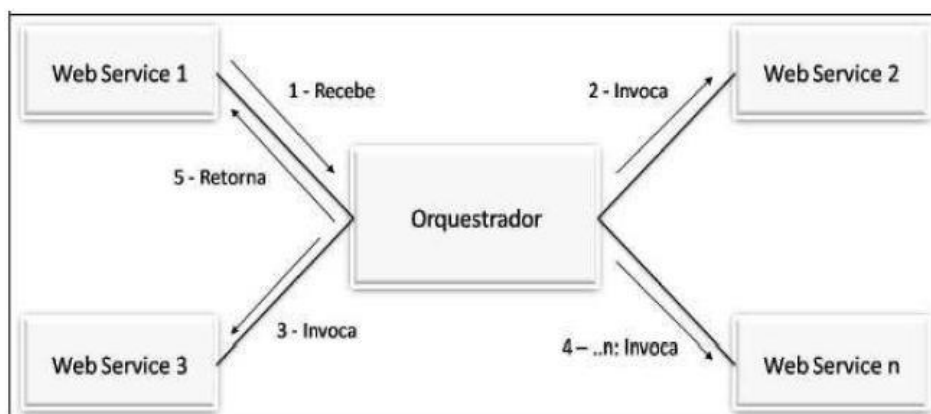
Para Fugita e Hirama (2012), a composição “é um conjunto de serviços agregados com o intuito de automatizar uma determinada tarefa complexa ou processo de negócio”. Ou seja, para um serviço ser considerado composto, pelo menos uma de suas operações deve invocar operações de outros serviços para compor sua lógica. Duas formas de implementar a composição de serviços, são: Coreografia e Orquestração.

Na composição por Orquestração as interações dos serviços utilizados no processo de negócio são gerenciadas por uma entidade controladora, também chamado de Orquestrador de Serviços. Empresas com vários sistemas interconectados ou processos de negócio trabalhando em conjunto necessitam de um elemento controlador para melhorar a interoperabilidade dos sistemas (PELTZ, 2003).

A utilização adequada da Orquestração pode reduzir consideravelmente a complexidade de implementação e manutenção dos sistemas, pois é possível unir processos de negócio sem necessitar que a regra de negócio seja desenvolvida novamente em outros sistemas.

A Figura 6 ilustra um exemplo do processo de uma Orquestração, onde um serviço, por exemplo, chamado Orquestrador, é invocado pelo *Web Service 1* que por sua vez requisita informações de outros serviços a fim de agrupá-las e retorná-las para o *Web Service 1*.

Figura 6: Processo de Orquestração de serviços.



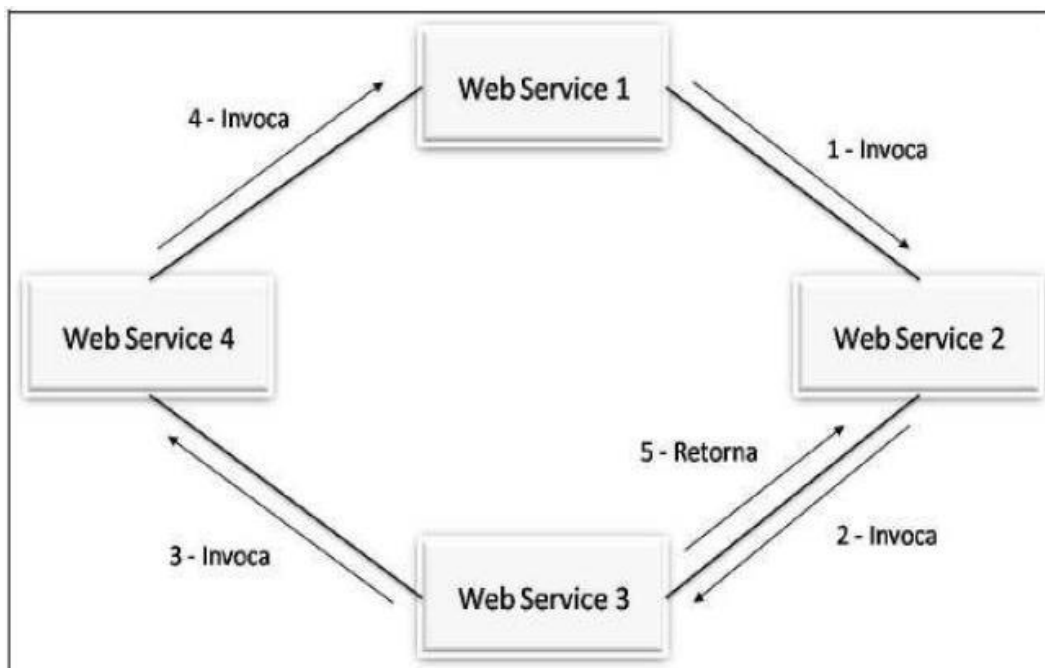
Fonte: Silva (2007).

Outra técnica muito utilizada na coordenação da execução de serviços é a Coreografia. Ao contrário da Orquestração, a Coreografia é mais colaborativa, permitindo que cada parte envolvida descreva seu papel na interação, trata da ordenação de mensagens e a interação sob a perspectiva de todas as partes, sem a necessidade de um elemento coordenador. Todos os serviços envolvidos na operação conhecem em que momento e com quem irão interagir, havendo uma cooperação, onde todos conhecem seus papéis dentro do fluxo (JOSUTTIS, 2008).

A Figura 7 representa uma composição de serviços fazendo uso da técnica de Coreografia. Cada *Web Service* invoca e recebe informações de outros *Web Services*, sendo assim, não há necessidade de um serviço controlador, porém todos têm que saber em que momento podem ou devem solicitar as informações administradas pelo outro elemento do processo.

A maior vantagem de utilizar a Orquestração ao invés da Coreografia é o ganho da flexibilidade, pois as inserções de novos serviços no processo de negócio podem ser realizadas de forma mais fácil e com menor impacto nos serviços já definidos, já que existe um responsável por controlar todas as requisições entre eles. A função de coordenador também auxilia um processo a estabelecer cenários alternativos e melhorar sua tolerância a falhas.

Figura 7: Coreografia de serviços.



Fonte: Silva (2007).

A Orquestração controla os nós (*web services*) e o fluxo de dados entre eles, ou seja, um processo central controla todos os envolvidos e coordena suas operações. Uma linguagem muito utilizada para definição e execução de um processo de negócio através da orquestração de serviços *web* é a BPEL.

2.6.3 Business Process Execution Language (BPEL)

Processos de negócio podem ser definidos em dois grupos: abstratos e executáveis. Processos de negócio abstratos são especificados sem maiores detalhes técnicos, pois não têm intenção de serem executados. Processos de negócio executáveis descrevem cada detalhe e aspecto do comportamento e interação das atividades entre seus participantes (OASIS, 2007).

BPEL surgiu com a junção de dois padrões distintos: *Web Services for Business Process Design* (XLANG) da Microsoft e *Web Service Flow Language* (WSFL) da IBM, e provê uma linguagem para especificação de ambos os tipos de processos (executáveis e abstratos) (CHANG, 2006).

BPEL define um modelo para descrever o comportamento dos processos de negócio baseado em interações entre processos e seus participantes. Essa interação é realizada através de interfaces de *Web Services*, encapsulados e ligados por uma conexão chamada de *partnerLink*. Com isso, é possível estabelecer múltiplas interações entre os participantes de forma coordenada para alcançar um objetivo de negócio. BPEL também introduz um mecanismo para lidar com exceções e falhas no processo e como essas falhas podem ser compensadas caso ocorram (OASIS, 2007).

Um processo BPEL é apresentado como um serviço, o qual compõe outros serviços, utilizando técnicas de composição. Alguns exemplos de atividades descritas no padrão WS-BPEL são:

- *Invoke*: chama um serviço web.
- *Receive*: aguarda a resposta de um cliente.
- *Reply*: gera uma resposta síncrona.
- *Assign*: manipula dados.

A estrutura geral de um processo de negócio especificado em BPEL possui quatro seções: *partner links*, variáveis, *fault handlers* e atividades. *Partner Link* é o canal pelo qual parceiros se comunicam através do mapeamento entre processo BPEL e o *port type* (porta de entrada de um serviço) do serviço; variáveis armazenam, formatam e transformam as mensagens trocadas entre parceiros; *fault handlers* são responsáveis por tratar as falhas que ocorrerem com a invocação de serviços de avaliação e de aprovação; atividades é a seção que

contém a descrição principal do processo, indica a ordem na qual os participantes da composição serão executados (OASIS, 2007). A Figura 8 contém um exemplo de código BPEL.

Figura 8: Exemplo de código BPEL.

```
<partnerLinks>
  <partnerLink
    name="SynchronousSample"
    partnerLinkType="ns1:partnerlinktype1"
    myRole="partnerlinktyperole1">
  </partnerLink>
</partnerLinks>
<variables>
  <variable name="outputVar"
    messageType="ns1:responseMessage"/>
  <variable name="inputVar"
    messageType="ns1:requestMessage"/>
</variables>
<sequence>
  <receive name="start" partnerLink="SynchronousSample" operation="operation1"
    portType="ns1:portType1" variable="inputVar" createInstance="yes"/>
  <assign name="Assign1">
    <copy>
      <from>concat('Hello ', $inputVar.inputType/ns2:paramA, '!!!')</from>
      <to>$outputVar.resultType/ns2:paramA</to>
    </copy>
  </assign>
  <reply name="end" partnerLink="SynchronousSample" operation="operation1"
    portType="ns1:portType1" variable="outputVar">
  </reply>
</sequence>
```

Fonte: Franceschini e Kon, 2010.

2.7 Considerações Finais do Capítulo

Ao longo deste capítulo foram apresentadas características importantes que ajudaram a definir que BPM, portanto, é um conceito proveniente da necessidade gerencial, o qual pode, e geralmente deve, ser apoiado pela TI a fim de auxiliar toda a gestão dos processos. Também foram apresentados alguns dos principais motivadores para adoção de uma gestão orientada a processos, como: necessidade de redução de custos, melhoria na qualidade dos produtos gerados, melhorar satisfação do cliente, aumentar competitividade, dentre outros.

Também foram abordadas algumas formas de automatizar processos de negócio, como por exemplo a utilização Sistemas de Gerenciamento de Processos de Negócio (BPMS), os quais auxiliam a gestão de todo o ciclo de vida de um processo. Outra forma bastante utilizada e eficiente para automação é a adoção de arquiteturas orientadas a serviços SOA, as quais facilitam a integração entre a área de negócio e a área de tecnologia da informação, promovendo uma maior flexibilidade e interoperabilidade entre os recursos disponíveis.

O próximo capítulo apresentará os conceitos referentes às abordagens dirigidas a modelos a fim de melhorar o entendimento da proposta de *framework* apresentada neste trabalho no capítulo 5.

CAPÍTULO 3

ABORDAGENS DIRIGIDAS A MODELOS

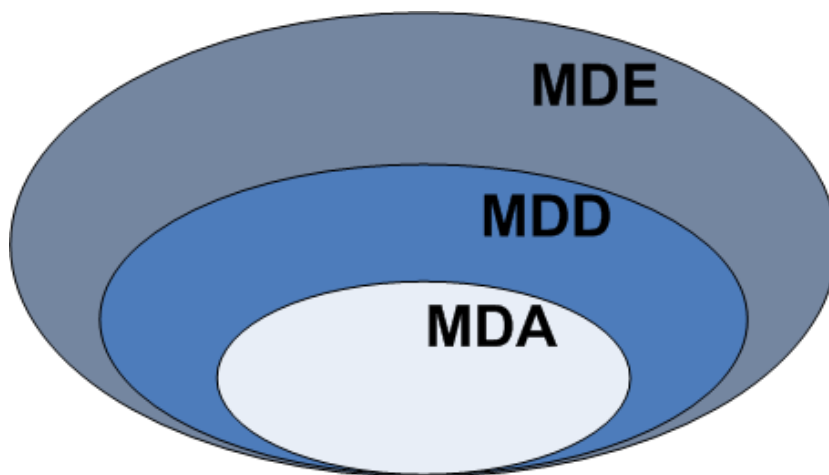
Neste capítulo serão abordados conceitos e características que definem Abordagens Dirigidas a Modelos, bem como a relação entre suas principais iniciativas. Também serão realizadas algumas considerações sobre transformações de modelos e as tecnologias envolvidas.

3.1 Visão Geral das Abordagens Dirigidas a Modelos

Nos últimos anos, o crescimento da complexidade no desenvolvimento de software aumentou e, um dos fatores apontados por France e Humpe (2007), é a distância entre o domínio do problema e o domínio da implementação da solução. Com isso, muitas empresas vêm investindo na industrialização do desenvolvimento de software, uma área em particular que se destaca é a *Model Driven Engineering* (MDE) ou Engenharia Dirigida por Modelos.

A MDE é um superconjunto se comparado a outros conceitos relacionados a iniciativas dirigidas a modelos como o *Model Driven Development* (MDD), pois ele engloba outras atividades do processo da Engenharia de Software. A *Model Driven Architecture* (MDA) é uma proposta de *framework* desenvolvida pela OMG, a qual representa uma visão da MDD (KLEPPE, WARMER e BAST, 2003). A Figura 9 ilustra a relação entre cada um desses conceitos.

Figura 9: Relação MDE, MDD, MDA.



Fonte: Demir (2006).

As próximas seções apresentarão mais detalhes a respeito de cada uma dessas iniciativas dirigidas a modelos: MDE, MDD e a MDA.

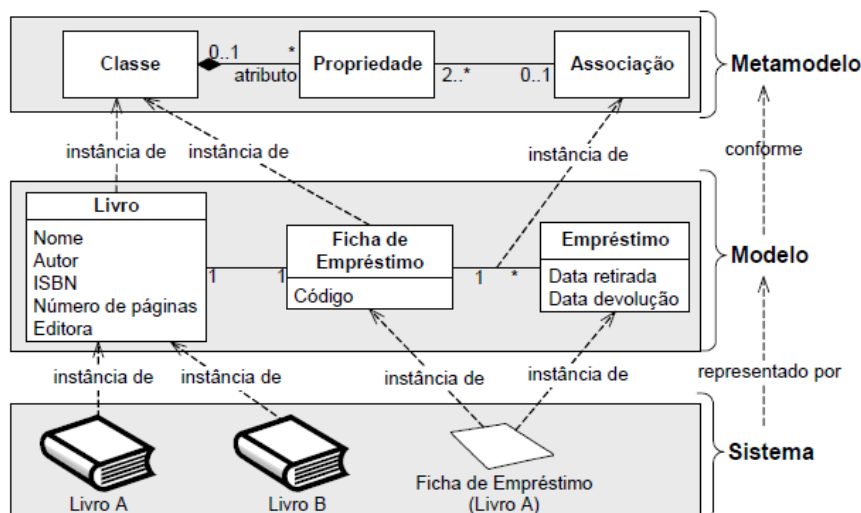
3.2 Model Driven Engineering (MDE)

O princípio básico da Engenharia Dirigida por Modelos é a ideia de que os artefatos principais no processo de desenvolvimento de software são os modelos, ao contrário do código, como é comumente destacado em outros paradigmas (MILICEV, 2009).

Para Siqueira (2011), de maneira geral, um modelo é uma representação simplificada da realidade criada para uma determinada finalidade. Ou seja, é uma abstração de um determinado cenário, o qual expõe apenas os detalhes relevantes para seu propósito. No contexto do desenvolvimento de software, Mellor *et al.* (2004) diz que um “modelo de um sistema é um conjunto de elementos que o descreve”. Para Kleppe, Warmer e Bast (2003), modelo é parte integrante de um sistema construído através de uma linguagem bem definida.

Em MDE, modelos são definidos a partir de seus metamodelos correspondentes. Vieira (2010) define metamodelo como “modelos que descrevem modelos”. Ou seja, metamodelos descrevem os mais variados tipos de componentes contidos no modelo e como eles são relacionados e restringidos (BÉZIVIN, 2006). Um exemplo de metamodelo é o apresentado na Figura 10, em que são apresentados alguns componentes de uma biblioteca mapeados em um modelo com base em um metamodelo UML. O modelo utiliza associações, classes e atributos para representar a relação entre livros, ficha de empréstimo e empréstimo, pois são elementos existentes no modelo que define sua sintática e semântica, ou seja, o metamodelo.

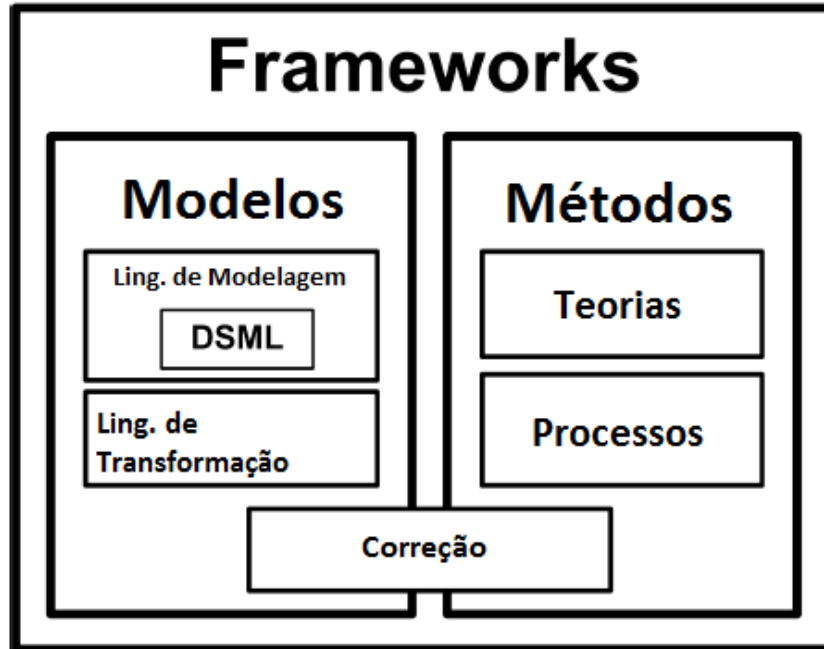
Figura 10: Modelo de domínio de uma biblioteca e seu metamodelo usado.



Fonte: Siqueira (2011).

Ameller (2009) destaca alguns pontos como sendo os principais tópicos de pesquisa em MDE (conforme ilustrado na Figura 11):

Figura 11: Principais tópicos de pesquisa em MDE.



Fonte: Traduzido de Ameller (2009).

- *Frameworks* e Ferramentas CASE: pesquisadores são muito dependentes dos recursos dessas ferramentas, pois geralmente não existe tempo suficiente para desenvolver todo o sistema.
- *Domain-Specific Modeling Language* (DSML): muito utilizada para pequenos e médios projetos, visto que normalmente ajudam a solucionar um problema em um domínio específico. O uso de uma DSML ajuda a reduzir erros de modelagem, uma vez que é possível validar os modelos com base na DSM utilizada.
- Metodologias: a complexidade do uso de abordagens dirigidas a modelos requer o uso de metodologias como forma de auxílio para resolver ou amenizar problemas.
- Linguagens de Transformações: tópico importante, pois todas as iniciativas dirigidas a modelos são limitadas de acordo com a expressividade de suas linguagens de transformações.

- Correção: como na MDE os modelos fazem parte do produto, ao invés de apenas da documentação, esforços para validar os modelos e garantir que estão corretos faz-se necessários.

3.3 *Model Driven Development (MDD)*

O primeiro artigo publicado oficialmente apresentando o paradigma MDD foi escrito por Mellor, Clark e Futagami (2003), os autores definiram *Model Driven Development* como uma abordagem a qual é possível construir um modelo de sistema e realizar transformações até conseguir algo concreto, como o código-fonte, por exemplo.

MDD oferece uma abordagem significativamente mais eficiente: modelos são abstratos e formais ao mesmo tempo. Abstração não quer dizer algo vago, mas sim relativo a essência de um problema, focado em detalhes relevantes para sua solução. A formalidade traz a possibilidade de gerar boa parte da implementação final de um software, não apenas classes e métodos estruturais. Alguns objetivos para adoção do paradigma MDD são abordados por Stahl *et al.* (2006), sendo eles:

- Incrementa velocidade de desenvolvimento. Alcançado através da automação, onde código pode ser gerado através de etapas de transformações a partir de modelos formalmente definidos.
- A automação das transformações e as linguagens de modelagem formalmente definidas permite aprimorar a qualidade, principalmente da arquitetura do software gerado, pois uma vez definido, poderá ser repetido uniformemente durante os incrementos futuros.
- As implementações de aspectos transversais podem ser alteradas em apenas um local, como por exemplo, nas regras de transformações. Esta separação de aspectos melhoram a manutenção de sistemas, evitando redundâncias e gerenciando mudanças tecnológicas.
- Permite um alto nível de reusabilidade, uma vez que a arquitetura, linguagem de modelagem e transformações tenham sido bem definidas, através da montagem de uma linha de produção de software.
- Melhora o gerenciamento da complexidade através da abstração, já que a modelagem permite ser descritos usando uma linguagem de modelagem orientada ao problema.

- Oferece produtividade nas áreas de tecnologias, engenharia e gerenciamento de software, através do uso de melhores práticas e processos.
- Finalmente, é possível obter interoperabilidade e portabilidade. Entretanto, estes benefícios são alcançados somente se existir uma padronização do paradigma, como por exemplo, o MDA da OMG (descrito com mais detalhes na seção seguinte), o qual separa a especificação da funcionalidade da especificação da plataforma.

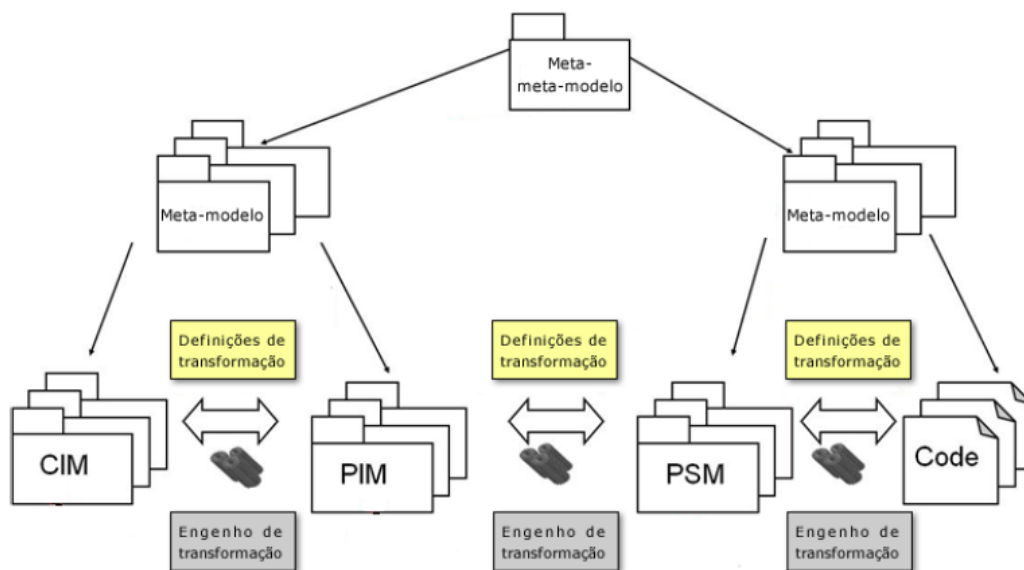
Em resumo, MDD é uma maneira de implementar softwares de forma rápida, eficiente e com o mínimo custo. Isso acontece porque modelos são menos ligados a forma de implementação tecnológica e mais próximos do domínio do problema. Sendo assim, são mais fáceis de especificar, compreender e manter, em alguns casos, chega a ser mais fácil de ser especificado por um especialista do domínio do problema do que por um especialista em tecnologia (SELIC, 2003).

3.4 Model Driven Architecture (MDA)

Uma das visões do *Model Driven Development* é a *Model Driven Architecture* (MDA), proposta pela OMG como um padrão de indústria independente de plataforma (VOS, 2008). De acordo com Santos (2011), em MDA existem três etapas com diferentes abstrações de modelos, conforme ilustradas na Figura 12, são elas:

- *Computational Independent Model* (CIM): responsável por descrever o domínio do problema, sem nenhum tipo de referência tecnológica. Preocupa-se em entender o problema do mundo real.
- *Platform Independent Model* (PIM): responsável por descrever o sistema, primeiro contato com o aspecto tecnológico através dos modelos, porém não existe uma plataforma específica em sua representação.
- *Platform Specific Model* (PSM): modelo com um maior nível de detalhamento em sua especificação. Proveniente do PIM, através de transformações. Nesta representação a plataforma será especificada, definida e modelada.

Figura 12: Etapas da arquitetura MDA.



Fonte: Ramalho (2008).

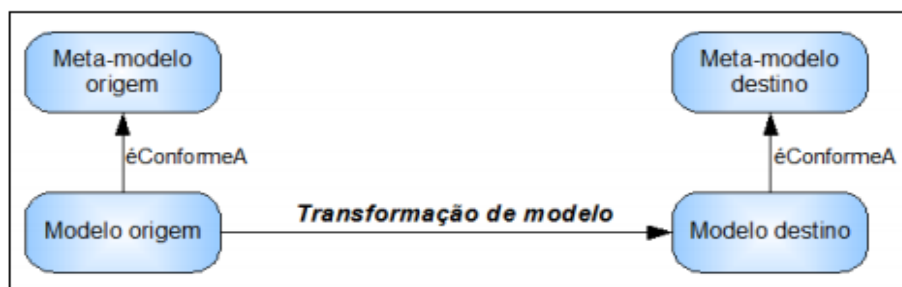
A Figura 12 indica que existe a possibilidade de transformações em qualquer direção, seja do CIM para o PIM ou PIM para CIM, PSM para código ou código para PSM. As gerações dos modelos são realizadas através de transformações, onde um ou mais modelos de origem podem ser transformados em um ou mais modelos destino. Mais detalhes sobre transformações serão apresentados a seguir.

3.5 Transformações de modelos

Uma transformação de modelos determina uma relação entre dois modelos que representam diferentes perspectivas ou diferentes níveis do sistema real modelado (MEDEIROS, 2009). Para Miller e Mukerji (2003), a transformação é, simplesmente, o processo de converter um modelo em outro modelo pertencentes ao mesmo sistema.

Conforme ilustrado na Figura 13, cada modelo é criado com base em um metamodelo, então uma linguagem de transformação é executada para transformar o modelo de origem no modelo destino. Uma linguagem de transformação utilizada em diversas abordagens dirigidas a modelos é a ATL.

Figura 13: Transformação de modelos.



Fonte: Medeiros (2009).

ATL *Transformation Language* é uma linguagem de transformação de modelos desenvolvida e integrada na plataforma Eclipse, a qual provê diversos recursos e padrões para facilitar sua utilização. A ATL disponibiliza formas de realizar transformações entre modelos, ou seja, pode ser aplicada em um determinado conjunto de modelos iniciais para gerar um outro conjunto de modelos resultantes (ATL, 2007).

ATL é uma linguagem híbrida, a qual permite realizar as transformações tanto com programação imperativa quanto declarativa. Com ATL existe a possibilidade de usar a programação declarativa para resolução de problemas mais simples e a imperativa para problemas mais complexos (CZARNECKI e HELSEN, 2006). A Figura 14 exemplifica uma regra de transformação implementada com ATL.

Figura 14: Regra ATL.

```

rule Member2Male {
  from
    s : Families!Member (not s.isFemale())
  to
    t : Persons!Male (
      fullName <- s.firstName + ' ' + s.familyName
    )
}
  
```

Fonte: (ATL, 2007).

Neste exemplo, presente na documentação ATL (ATL, 2007), o modelo origem “*Families*” deverá gerar o modelo destino “*Persons*”. Para tanto, uma das regras deve ser identificar o nome da família no modelo “*Families*” para compor o nome completo da pessoa no modelo “*Persons*”. Esta regra também valida se a pessoa criada no destino deve ser homem ou mulher (com base na especificação do modelo origem).

3.6 Considerações Finais do Capítulo

Neste capítulo foram apresentados os conceitos referentes às abordagens dirigidas a modelos, incluindo: MDE, MDD e MDA. Todos compartilham da ideia de que modelos são os artefatos principais no desenvolvimento de software, ao invés do código fonte, como é colocado pelo paradigma tradicional de desenvolvimento de software.

Dentre os diversos benefícios apontados pelas iniciativas dirigidas a modelos, destacam-se: o aumento da produtividade e eficiência na construção de novas soluções, melhoria na qualidade do software gerado, bem como a portabilidade e interoperabilidade alcançadas com o uso de padrões de abordagens dirigidas a modelos, como por exemplo, o MDA.

Diante desse contexto, com base no objetivo apresentado neste trabalho, fez-se necessário estudar como o uso de abordagens dirigidas a modelos podem auxiliar no desenvolvimento de soluções baseadas em processos e arquiteturas orientadas a serviços.

No próximo capítulo serão apresentados os resultados de uma Revisão Sistemática de Literatura, a qual buscou identificar as soluções que integrassem os conceitos de BPM e SOA na perspectiva da Engenharia Dirigida a Modelos.

CAPÍTULO 4

BPM E SOA NA PERSPECTIVA DO DESENVOLVIMENTO ORIENTADO A MODELOS

Nos capítulos anteriores, foram abordados definições, conceitos e os diversos benefícios tanto na adoção de uma Gestão de Processos de Negócio, quanto no desenvolvimento de soluções utilizando abordagens dirigidas a modelos. Sendo assim, através de uma Revisão Sistemática de Literatura, buscou-se levantar as propostas existentes de Abordagens Dirigidas a Modelos envolvendo os conceitos de BPM e SOA. São apresentados neste capítulo, o protocolo, as *strings* de busca e a análise dos resultados obtidos por meio da Revisão Sistemática de Literatura.

4.1 Protocolo da Revisão Sistemática

Esta seção apresenta a descrição do protocolo utilizado na aplicação desta revisão sistemática. Com o intuito de avaliar estudos envolvendo as abordagens dirigidas a modelos na gestão de processos e arquiteturas SOA, foram formuladas as seguintes perguntas:

1. Quais abordagens dirigidas a modelos utilizando os conceitos SOA e BPM estão sendo desenvolvidas?
2. Quais aplicabilidades demonstradas através de experimentos ou estudos de casos?

Os critérios de seleção das fontes foram:

- Disponibilidade de consulta de artigos através da web.
- Utilização de mecanismos de buscas através de conjuntos de palavras-chaves.
- Artigos escritos na língua inglesa.

Conforme demonstrado na Figura 15, após a execução das *Strings* de buscas, foram selecionados todos os artigos com data de publicação entre 2010 e 16 de outubro de 2015 (data da última execução do protocolo desta revisão sistemática). Em seguida, foram selecionados apenas artigos escritos na língua inglesa para leitura dos resumos e introdução. Na etapa seguinte, os artigos que não abordaram conceitos de SOA e BPM dentro do contexto da MDE,

ou apresentaram apenas levantamentos do estado da arte, foram descartados. Por fim, foi realizada a leitura completa dos artigos restantes.

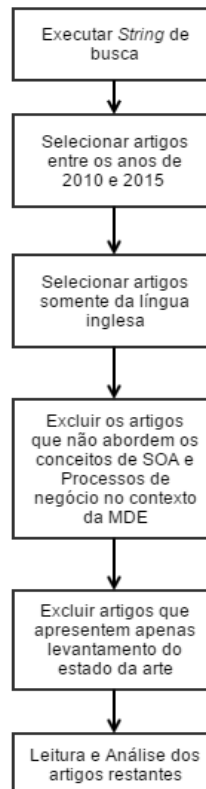
As consultas ocorreram nas bases IEEE Explorer e ACM Digital Library. Na IEEE, a seguinte *string* de busca foi utilizada:

- *((("Abstract":"MDD" OR "MDE" OR "Model-driven development" OR "Model driven Engineering") AND (p_Abstract:SOA OR "Service Oriented Architecture") AND (p_Abstract:"BPM" OR "Business Process Management" OR "Business Process" OR "Business Process Modeling Notation"))).*

Na base ACM Digital Library, a *string* utilizada foi:

- *((((Abstract:MDD) OR (Abstract:"Model Driven Development") OR (Abstract:MDE) OR (Abstract:"Model Driven Engineering"))) AND ((Abstract:SOA) OR (Abstract:"Service Oriented Architecture"))) AND ((Abstract:BPM) OR (Abstract:"Business Process Management") OR (Abstract:"Business Process") OR (Abstract:"Business Process Modeling Notation"))).*

Figura 15: Protocolo da revisão sistemática.



Fonte: Autores (2015).

Em seguida, serão exibidos os resultados da execução do protocolo da revisão sistemática de literatura apresentado anteriormente, apresentando os quantitativos de artigos encontrados em cada etapa.

4.2 Resultados da Revisão Sistemática

Após aplicação do protocolo da revisão sistemática definido na seção anterior, foram obtidos 07 artigos como resultado final. O Quadro 8 apresenta o quantitativo de artigos encontrados após aplicação de cada critério do protocolo.

Quadro 8: Resultados do processo de seleção.

Métodos do Protocolo	IEEE	ACM	Total
Execução da <i>string</i> de busca (ano 2010 - 2015)	17	8	25
Selecionados apenas escritos na língua inglesa	16	8	24
Após exclusão de artigos que não abordaram SOA e BPM no contexto da MDE	8	4	12
Após exclusão de artigos que realizaram apenas o levantamento do estado da arte	8	4	12
Após exclusão dos artigos repetidos	3	4	7

Fonte: Autores (2015).

Os trabalhos selecionados foram após os critérios de exclusão e inclusão foram:

- (DELGADO, GUZMÁN e PIATTINI, 2010), (DAHMAN, CHAROY e GODART, 2010), (DAHMAN, CHAROY e GODART, 2011), (DAHMAN, CHAROY e GODART, 2013), (MUSTAFA e BOCHMANN, 2011), (SINDHGATTA, 2013) e (SOSA-SANCHEZ *et al.*, 2014).

Os resumos dos trabalhos selecionados para análise desta revisão sistemática de literatura são apresentados a seguir.

4.3 Resumos dos Trabalhos

Delgado, Guzmán e Piattini (2010) desenvolveram o *framework* MINERVA, cujo objetivo é prover uma solução integrada para desenvolvimento de processos de negócio orientado a serviços através de transformações de modelos BPMN para SoaML. Os autores realizaram uma prova de conceito utilizando diversos plug-ins no Eclipse para demonstrar as etapas de uma transformação de um processo hospitalar como exemplo.

No artigo apresentado por Dahman, Charoy e Godart (2010), os autores apresentam uma abordagem de Desenvolvimento Dirigido a Modelos (MDD), cujo objetivo é automatizar a produção de ativos de TI alinhados aos processos de negócio da organização. Para tanto, os autores propõem a especificação e implementação de regras de mapeamento para realizar a transformação entre dois metamodelos: *Business Process Modeling Notation* (BPMN) e *Service Component Architecture* (SCA).

O trabalho realizado por Mustafa e Bochmann (2011) utiliza a ferramenta da IBM, a *Rational Software Architect* (RSA), para transformar diagramas de atividades da UML em especificações WSDL/BPEL de componentes de *Web Services* em arquiteturas SOA. Dessa forma, os autores identificam as principais dificuldades apresentadas pela ferramenta e discutem como resolvê-las realizando simples modificações nos componentes gerados em BPEL.

Sindhgatta (2013) propõe um *framework* baseado em modelos, a qual permite executar composições de serviços em tempo de projeto (com objetivo de realizar simulações). Os modelos de processo e serviços são transformados em modelos executáveis de UML, utilizando a *UML Action Language*. A abordagem foi demonstrada através de 17 processos de negócio e 127 atividades utilizando 13 interfaces de serviços.

A proposta apresentada por Sosa-Sanchez (2014) auxilia o gerenciamento das mudanças contínuas nos processos de negócio de uma empresa. Para tanto, é proposta uma abordagem dirigida a modelos sistemática e semiautomatizada para atualizar e inserir os *Legacy Web Applications* (LWA) em uma arquitetura SOA. A abordagem utiliza técnicas de reengenharia para recuperar os modelos dos sistemas legados existentes bem como modelos que identifiquem os serviços, em seguida os serviços descobertos são alinhados aos processos de negócios mapeados, os quais servem de entrada para transformações e gerações de modelos orquestrados pelo BPEL (*Business Process Execution Language*).

Dahman, Charoy e Godart (2011) entendem que, na prática, a evolução de um projeto de software dirigido a modelos é mais desafiadora que seu início. Dessa forma, é apresentado

no artigo uma abordagem incremental de transformações, cujo objetivo é atualizar os modelos de serviços gerados quando os processos de negócio da organização evoluem, mantendo a consistência estrutural de sua arquitetura. Em um trabalho futuro, realizado por Dahman, Charoy e Godart (2013), os autores desenvolvem um protótipo para validar a consistência da abordagem anteriormente proposta.

4.4 Análise dos Resultados

Observa-se, por meio desta revisão sistemática de literatura, que as publicações descrevendo abordagens dirigidas a modelos no contexto de BPM e SOA estão evoluindo, mas ainda encontram diversos obstáculos. A abstração necessária para realizar transformações entre modelos em um alto nível de abstração, como é o caso de BPM e SOA, exige um grau de complexidade maior quando comparado a transformações utilizando abordagens mais dependentes de plataformas e mais próximas do código.

Dessa forma, em geral, os artigos que realizaram estudos de casos ou experimentos para comprovação de suas hipóteses, mencionaram que a complexidade imposta pelo domínio do problema influencia no sucesso de suas abordagens. Ou seja, ainda existem problemas para lidar com processos extensos e complexos. O Quadro 9 apresenta algumas informações sobre os artigos selecionados.

Quadro 9: Comparativo dos artigos selecionados.

Artigo	Metamodelo(s) inicial(is)	Metamodelo após transformações	Método ou linguagem de transformação utilizada	Tecnologias utilizadas
Delgado, Guzmán e Piattini (2010)	BPMN	SoaML	QVT	Medini QVT plug-in Eclipse, BPMN Modeler Plug-in Eclipse, MagicDraw Cameo SOA + plug-in no Eclipse
Dahman, Charoy e Godart (2010)	BPMN	SCA assembly model	ATLAS	Eclipse Modeling Framework
Dahman, Charoy e Godart (2011)	BPMN	SCA assembly model	Não mencionado	Implementado em Java como uma extensão do Yaoqiang BPMN Editor (baseado em Jgraph Library)

Artigo	Metamodelo(s) inicial(is)	Metamodelo após transformações	Método ou linguagem de transformação utilizada	Tecnologias utilizadas
Dahman, Charoy e Godart (2013)	BPMN	SCA assembly model	Não mencionado	Não mencionadas
Mustafa e Bochmann (2011)	UML	BPEL	Não mencionado	IBM Rational Software Architect (RSA) e IBM WebSphere (executar o processo gerado)
Sindhgatta (2013)	UML	UML Action Language	APIs providas pela EMF-implementation UML2 para Eclipse	IBM Rational Software Architect (RSA), Alf cod Generator
Sosa-Sanchez (2014)	BPMN e SoaML	Serviços (SoaML) alinhados aos processos de negócio (BPMN)	Algoritmo Wang-Ali (descobertas de similaridade ontológica entre dois conceitos)	Não mencionadas

Fonte: Autores (2015).

Pode-se perceber, por exemplo, que as propostas apresentadas partiram das notações BPMN ou UML nos modelos iniciais. O uso de BPMN tem como característica ser flexível e contar com uma maior semântica para criar modelos próximos ao domínio do problema e do usuário, enquanto que a UML pode ter um nível de abstração mais baixo, fazendo com que seja mais fácil realizar procedimentos de transformações em níveis mais técnicos e próximos ao código.

Outro ponto a ser destacado é que maioria dos trabalhos não apresentam uma solução completa para uma abordagem dirigida a modelos utilizando BPM e SOA. As propostas abrangem pequenas etapas ou atividades de transformações de modelos sem se preocupar com as etapas seguintes. O *framework* MINERVA, proposto por Delgado, Guzmán e Piattini (2010), apesar de mapear o processo de desenvolvimento dirigidos a modelos como um todo, não discute alguns pontos importantes, como por exemplo: a análise de sistemas legados, ou seja, o levantamento dos possíveis serviços providos pelos sistemas existentes; a evolução dos processos transformados devido a mudanças de regras de negócio; e o uso de DSMLs para uma melhor especificação dos modelos, bem como para realizar suas validações.

4.5 Considerações Finais do Capítulo

Este capítulo apresentou uma revisão sistemática de literatura com objetivo de buscar propostas de abordagens dirigidas a modelos baseadas em processos e arquiteturas orientadas a serviços, identificando suas principais aplicações, notações (metamodelos) e tecnologias envolvidas.

Os trabalhos selecionados apresentaram diferentes formas de integrar Gerenciamento de Processos de Negócio e Arquiteturas orientadas a Serviços, utilizando diferentes representações de modelos e abordagens de transformações. Contudo, o objetivo sempre converge para desenvolver soluções orientadas a processos com uma maior produtividade e eficiência para, assim, garantir que os objetivos da organização sejam alcançados.

Entretanto, notou-se a falta de propostas de *frameworks* que contemplem todas as etapas de transformações propostas pela arquitetura MDA. Em geral, os trabalhos levantados dedicavam-se a uma determinada etapa e não demonstravam o funcionamento completo das abordagens, como a geração do modelo CIM até o resultado final concreto.

Nesse contexto, o próximo capítulo apresentará o BPM4Services, *framework* dirigido a modelos, cujo objetivo é aumentar a eficiência no desenvolvimento de soluções orientadas a processos.

CAPÍTULO 5

FRAMEWORK BPM4SERVICES

Conforme visto nos resultados da revisão sistemática de literatura, existem diversas lacunas e dificuldades encontradas nas propostas de abordagens dirigidas a modelos utilizando conceitos de SOA e BPM, por exemplo: falta de trabalhos que apresentem uma solução completa de automação de processos, descrevendo todo o ciclo de vida e etapas baseado na engenharia dirigidas a modelos; também existem dificuldades no processo de transformação, visto que alguns elementos originais BPMN carecem de uma semântica mais específica e detalhada para uso nas regras de mapeamento; por fim, nenhum dos trabalhos encontrados apresentam o uso de testes automatizados para validação dos modelos e códigos gerados pelo processo de transformação.

Nesse contexto, desenvolveu-se o *framework* BPM4Services, o qual propõe abordar uma solução completa e dirigida a modelos para automação de processos de negócios utilizando uma Arquitetura Orientada a Serviços. O BPM4Services também apresenta uma perspectiva de testes automatizados para os modelos e serviços criados, bem como uma extensão do metamodelo BPMN v2.0 a fim de obter uma semântica mais precisa para utilizar no processo de transformação.

Este capítulo discute a estrutura e características do *framework* BPM4Services, apresentando como seus principais artefatos se relacionam e como acontecem as transformações entre seus modelos.

5.1 Estrutura do BPM4Services

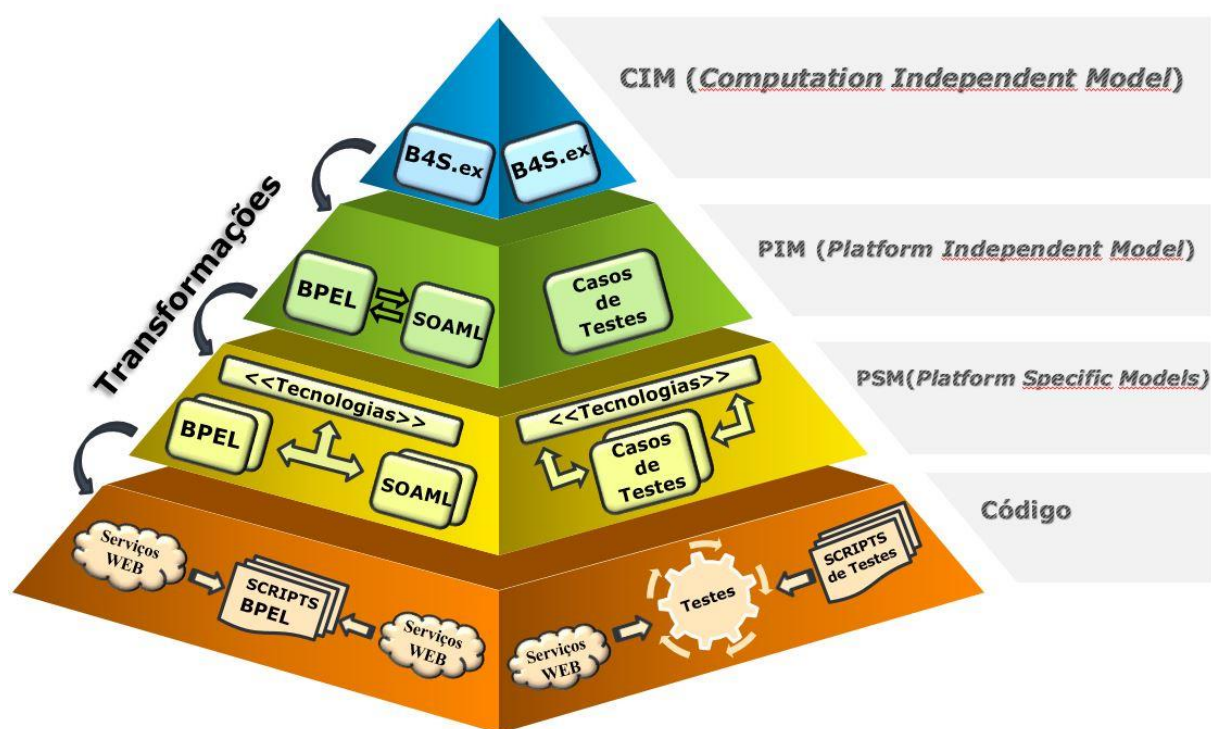
Segundo Johnson (1991), um *framework* é um conjunto de elementos unidos com o objetivo de alcançar um conjunto de responsabilidades para uma determinada aplicação ou domínio de problema. Um *framework* também deve definir a estrutura geral das aplicações que fazem parte de sua composição, bem como a forma como seus componentes se relacionam e quais seus mecanismos para reutilização (BARROS, 2009). Em outras palavras, um *framework* é um modelo que guia o usuário para entender quais e quando determinados artefatos serão gerados em um cenário específico (DRAFFIN, 2007).

De acordo com Barros (2009), de modo geral, um *framework* aumenta a eficiência do processo de desenvolvimento de software, principalmente se utilizado em ambientes heterogêneos, compartilhado por diferentes equipes ou no desenvolvimento de múltiplas aplicações dentro de um mesmo contexto.

Dessa forma, para construção do BPM4Services, foi utilizado o conceito de *framework*, visto que, em ambientes orientados a processos, características como interoperabilidade e portabilidade são fatores cruciais e determinantes para o sucesso de uma solução. O *framework* também é responsável por definir a estrutura geral e a forma como seus componentes se relacionam sob a perspectiva de uma solução dirigida a modelos para automação de processos de negócio.

O BPM4Services foi concebido com base na arquitetura MDA da OMG, uma vez que sua estrutura é composta pelas 4 camadas que a MDA propõe: *Computation Independent Model* (CIM), *Platform Independent Model* (PIM), *Platform Specific Models* (PSM) e Código. A Figura 16 apresenta a estrutura completa do BPM4Services.

Figura 16: Estrutura do BPM4Services.



Fonte: Autores (2016).

Conforme apresentado na Figura 16, o BPM4Services possui basicamente duas perspectivas de evolução, a primeira é responsável pela construção dos modelos que

especificam os serviços criados e orquestrados pelo BPEL, enquanto que a segunda perspectiva apresenta a criação da automação dos testes destes serviços.

No topo da estrutura encontra-se o *Computation Independent Model* (CIM), também conhecido como modelo de domínio ou modelo de negócio, o qual define as regras de negócio representadas no “mundo real” e não em modelos de sistema. Para representação do CIM, foi escolhido o padrão de modelagem BPMN pois, conforme visto no capítulo 2, é uma notação compreensível por todos os envolvidos em um ambiente orientado a processos, desde os analistas de negócio até os desenvolvedores responsáveis por implementar tecnologias que irão suportar a execução dos processos. Ou seja, BPMN é um padrão para modelagem, o qual utiliza técnicas e componentes bem definidos voltados especialmente para definição e documentação de processos de negócio. Contudo, no contexto de uma transformação de modelos, o BPMN carece de alguns componentes com uma semântica mais específica, dessa forma, foi proposta uma extensão do metamodelo BPMN para ser utilizada neste *framework* e em suas transformações. A extensão, denominada B4S.ex, especifica novos componentes e cria novas propriedades, as quais são utilizadas nas regras de mapeamento executadas para geração automática dos demais modelos do BPM4Services. O modelo especificado com base no metamodelo B4S.ex é o ponto de partida para geração dos modelos subsequentes nas duas perspectivas do *framework*.

Na camada seguinte, encontra-se a *Platform Independent Model* (PIM), um modelo de alto nível de abstração que contempla a especificação de funcionalidades e restrições de um sistema, porém independente de qualquer tecnologia. Nesta camada do *framework* são gerados os modelos de casos de testes com base nas entradas e saídas dos serviços mapeados no processo de negócio, bem como o modelo BPEL, responsável por descrever o comportamento dos processos de negócio baseado em suas interações entre processos e seus participantes através de interfaces de serviços que, por sua vez, são gerados a partir de uma especificação SoaML. SoaML provê uma notação para arquiteturas e modelos baseados em SOA como uma extensão do padrão UML, permitindo projetar a interoperabilidade e integração a nível de modelos.

A camada *Platform Specific Models* (PSM) é representada pelos mesmos metamodelos e especificações da camada PIM, porém adicionando detalhes da tecnologia utilizada. Pelo BPM4Services se tratar de um *framework*, o qual não determina nem associa tecnologias específicas na composição de sua estrutura, a tecnologia fica a critério de quem implementar este *framework*, assim como a construção de diferentes regras de mapeamentos para diferentes tecnologias utilizadas. Ou seja, o BPM4Services é um *framework* flexível, visto que sua estrutura permite escolher diferentes tecnologias apenas alterando a camada PSM, sem ser

necessário alterar as camadas superiores, basta que sejam construídas novas regras de transformações a partir dos modelos PIM gerados.

Por fim, na camada mais inferior, encontra-se o código que, nesta proposta de automação de processos de negócio, são as implementações dos serviços, os scripts de execução BPEL e os scripts de testes. Nesta camada os modelos trocam informações entre si, pois os scripts de testes executam os serviços para que possam analisar as diferentes saídas com base nas diferentes entradas e validar sua funcionalidade. Por outro lado, o script BPEL localiza os serviços criados através dos modelos SoaML e realiza sua orquestração através das chamadas de suas interfaces. Havendo qualquer problema nos testes automatizados, o responsável pelo processo é notificado e pode ser necessário ajustar os modelos gerados ou mesmo o modelo B4S.ex especificado no início do processo.

Para realizar as transformações entre as camadas apresentadas, optou-se pela linguagem de transformação ATL, por ser uma linguagem híbrida, a qual permite realizar as transformações tanto com programação imperativa quanto declarativa. Com ATL existe a possibilidade de usar a programação declarativa para resolução de problemas mais simples e a imperativa para problemas mais complexos (CZARNECKI e HELSEN, 2006). De acordo com os estudos de Bosems (2011) e Amstel *et al.* (2011), quando há um aumento da complexidade nos modelos de entrada, as transformações utilizando ATL são executadas até 5 vezes mais rápidas que outras linguagens de transformação, além de uma maior facilidade em lidar com vários modelos expressados em diferentes linguagens e tecnologias.

O *framework* BPM4Services contempla todo o ciclo de vida de automação de um processo sob a perspectiva da Engenharia Dirigida a Modelos. As seções subsequentes descrevem a criação do B4S.ex, extensão do metamodelo BPMN, detalhando a metodologia escolhida e utilizada para especificá-lo, assim como a definição das regras de mapeamento utilizadas na transformação de um modelo B4S.ex para um modelo SoaML.

5.2 Extensões BPMN

Conforme visto no capítulo 2, a notação BPMN é um padrão de modelagem de processos de negócio bastante difundido no mercado, principalmente devido à sua expressividade, seu metamodelo bem definido e sua alta capacidade de integração. Um dos recursos BPMN 2.0 prevê um mecanismo de extensão, o qual permite anexar novos componentes ou atributos aos seus elementos originais. Mecanismos de extensão permitem representar conceitos de linguagens específicas de domínio em linguagens de modelagem com

propósito mais geral, assim como podem ser utilizados para transformar modelos conceituais em modelos específicos de plataforma ou código (OMG, 2011).

A seguir, serão apresentadas algumas propostas de extensões do modelo BPMN encontradas e desenvolvidas entre os anos de 2010 e 2014. No Quadro 10, algumas características foram destacadas em cada proposta analisada, foram elas: o escopo da aplicação da extensão, a utilização de alguma metodologia que direcionasse o desenvolvimento de uma extensão, o uso do mecanismo de extensão proposto pela versão 2.0 do BPMN, a apresentação do metamodelo da extensão e, por fim, se a extensão proposta prevê algum tipo de transformações entre modelos.

Fonseca, Oliveira e Pereira (2011) propõem uma extensão do BPMN incrementando características de *tailoring* presentes no *Software & Systems Process Engineering Meta-Model 2.0* (SPEM), o qual é um metamodelo baseado no MOF 2.0 - *Meta Object Facility* utilizado para adaptar Processos de Desenvolvimento de Software (PDS) a fim de atender às necessidades de diferentes projetos ou organizações. O BPMNt, como foi chamado pelos autores, utilizou as classes disponíveis no modelo BPMN 2.0 para realização de extensões do seu metamodelo, sendo assim, criou-se novos atributos adicionados as classes existentes de forma a criar a extensão proposta.

Braun *et al.* (2014) propõem uma adaptação do modelo BPMN para representação de processos clínicos. De acordo com os autores, o BPMN consegue cobrir diversos requisitos dos processos clínicos, porém falha ao representar suas restrições e condições especiais. Dessa forma, com o objetivo de suprir essas lacunas, foi desenvolvido o BPMN4CP a partir da metodologia apresentada por Stroppi *et al.* (2011), o qual define uma série de etapas para criação de uma extensão do BPMN através das classes já disponíveis na versão 2.0 do metamodelo. Os autores também projetaram uma ontologia que mapeou a equivalência entre os conceitos do domínio e elementos do BPMN.

Goldner e Papproth (2011) propõem uma extensão da sintaxe BPMN para representar restrições legais explicitamente dentro do modelo. Como consequência, também estenderam o editor BPMN para exportar as restrições legais em forma de requisitos para uma ferramenta de gerenciamento de requisitos. Os requisitos legais são mapeados em novos artefatos no modelo, chamados: "*Regulation*" e "*Regulation Group*", os quais são compostos por atributos que ajudam a identificar as referências das leis (nome da lei, texto da lei, artigo, parágrafo e organização). O editor BPMN foi o ORXY da Hasso-Plattner-Institute, o qual permite sua extensão através de plug-ins. O gerenciador de requisitos foi o Erudine Behavior Engine da Erudine.

Gao, Derguech e Zaremba (2011) apresentaram a proposta do *Structured Web Resource* (SWR), extensão do BPMN para permitir a ligação de seus elementos com entidades externas seguindo os conceitos de *Linked Data*, o qual é um conjunto de melhores práticas para publicar e interconectar estruturas de dados na Web. Nessa extensão utilizou-se as diferentes perspectivas apresentadas pela *Architecture of Information System* (ARIS): visão organizacional, funcional, de dados, controle e produto/serviço.

Saeedi, Zhao e Sampaio (2010) propõem incorporar requisitos de qualidade em modelos BPMN a partir de uma extensão proposta. Dessa forma, a extensão poderia auxiliar a escolha dos diferentes serviços disponíveis. A abordagem focou nos principais requisitos contemplados pelo *Service Level Agreement* (SLA), são eles: custo, tempo e confiabilidade.

Friedenstab et al. (2012) apresentam uma abordagem para modelar os principais conceitos de *Business Activity Monitoring* (BAM) a partir da extensão do BPMN. Dessa forma, pretende-se expressar diferentes categorias de *Key Performance Indicators* (KPIs), as quais não são contempladas pelo metamodelo BPMN. Os principais resultados foram: um metamodelo que, junto com a descrição textual dos construtores semânticos, define os aspectos conceituais da extensão; símbolos gráficos que representam a sintaxe abstrata da abordagem; e um cenário que demonstra a aplicação prática da linguagem.

Supulniece, Businska e Kirikova (2010) descrevem uma proposta de extensão do BPMN para adicionar a perspectiva do gerenciamento de conhecimento, sendo possível expressar diferentes aspectos do conhecimento, informação e dados aos modelos de processos de negócio.

A utilização de classes nativas do BPMN para criar novas extensões permite uma maior integração com ferramentas de gerenciamento de processos de negócio, visto que é necessária uma especificação formal da extensão em concordância com o metamodelo BPMN. Ou seja, as novas extensões são mais facilmente interpretadas e integradas com ferramentas BPM existentes, pois os novos componentes seguem as mesmas regras e normas dos demais componentes do metamodelo original.

No Quadro 10, foi possível observar que, dentre os trabalhos encontrados, quatro utilizaram o mecanismo de extensão com as classes nativas do BPMN 2.0, contudo, somente um trabalho fez uso de uma metodologia formal para definição da extensão proposta utilizando todas as normas previstas no mecanismo de extensão BPMN. Os demais trabalhos apresentaram apenas componentes gráficos e explicaram sua nova semântica dentro do modelo de processo de negócio. Nenhum deles propôs uma extensão para modelagem de serviços, ou apresentou estratégias ou regras de transformações entre modelos em suas abordagens.

Quadro 10: Comparativo entre as propostas de extensões BPMN.

Autores	Escopo de aplicação da extensão	Utiliza metodologia?	Apresenta o metamodelo da extensão?	Utiliza classes de extensão do BPMN 2.0?	Utilizados para transformações de modelos?
Fonseca, Oliveira e Pereira (2011)	Processos de desenvolvimento de software	Não	Sim	Sim	Não
Braun <i>et al.</i> (2014)	Protocolos clínicos	Sim	Sim	Sim	Não
Goldner e Papproth (2011)	Restrições legais em artefatos do modelo	Não	Não	Não	Não
Gao, Derguech e Zaremba (2011)	Estruturas de dados e suas conexões com conceitos de <i>Linked Data</i>	Não	Não	Sim	Não
Saeedi, Zhao e Sampaio (2010)	Requisitos de qualidade	Não	Sim	Sim	Não
Friedenstab <i>et al.</i> (2012)	Gerenciamento de processos de negócios e indicadores de desempenho	Não	Sim	Não	Não
Supulniece, Businska e Kirikova (2010)	Gerenciamento de conhecimento	Não	Não	Não	Não

Fonte: Autores (2016).

Nesse contexto, este trabalho também propõe a criação do B4S.ex, uma extensão BPMN, a qual utilizará novas características e elementos para representar com mais detalhes os serviços previstos no processo de negócio. Dessa forma, a partir da utilização da nova semântica dos componentes que representarão os serviços e suas interações, será possível transformar o modelo BPMN em modelos SoaML, apresentando os serviços dentro de uma arquitetura SOA através de diagramas mais formais e representativos, os quais serão entradas para novas transformações até a geração do código.

Por tanto, para criação do B4S.ex foi utilizada a metodologia proposta por Stroppi, Chiotti e Villarreal (2011), a qual define uma série de passos e regras para definição de uma extensão válida dentro dos padrões do metamodelo BPMN.

5.2.1 Metodologia

Stroppi, Chiotti e Villarreal (2011) e Braun *et al.* (2014) acreditam que, embora os elementos sejam bem definidos, um dos fatores pela falta de uso do mecanismo de extensão proposto no BPMN 2.0 é a falta de um guia metodológico para direcionar a criação da extensão

com base nas regras do metamodelo. Stroppi, Chiotti e Villarreal (2011) adicionam como problema a falta de uma notação gráfica para representar os modelos das extensões, identificando o papel de cada elemento criado.

Stroppi, Chiotti e Villarreal (2011) descrevem uma metodologia para criação de extensões BPMN baseada em MDA. Os autores utilizam a UML para representar graficamente o metamodelo BPMN, a extensão proposta e suas transformações em documentos XML que podem ser processados por ferramentas BPMN.

A metodologia proposta consiste em quatro etapas, as quais são apresentadas abaixo:

1. Definição do modelo conceitual chamado *Conceptual Domain Model of the Extension* (CDME), representado através da UML.
2. Definição do modelo BPMN *plus Extensions* (BPMN +X), o qual descreve a extensão a partir dos mecanismos da extensão propostos no BPMN.
3. Transformação do modelo BPMN+X em um XML *Schema Extension Definition Model*.
4. Transformação do XML *Schema Extension Definition Model* em um XML *Schema Extension Definition Document*.

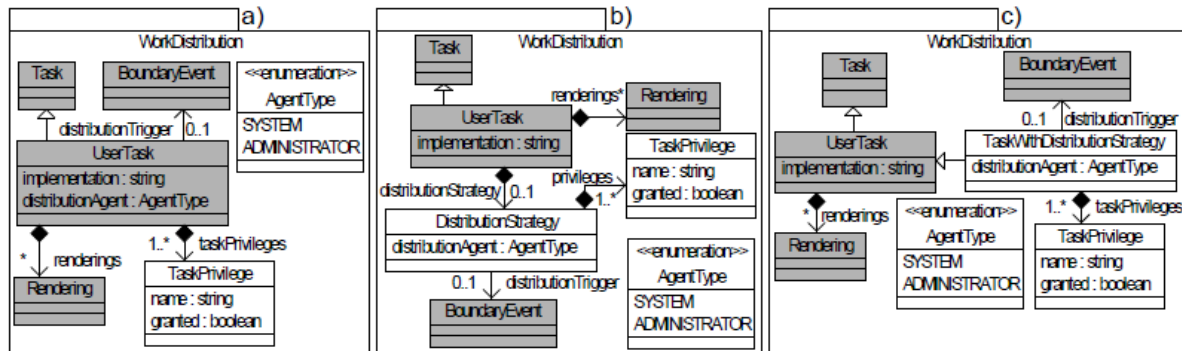
Para o desenvolvimento deste trabalho, utilizou-se apenas as etapas 1 e 2 da metodologia proposta, visto que o objetivo principal do desenvolvimento desta extensão é criar elementos que possam facilitar as transformações dos serviços modelados no processo de negócio (BPMN) em modelos de arquitetura orientadas a serviços (SoaML).

A primeira etapa determina a criação do CDME, o qual descreve os conceitos do domínio que serão representados pela extensão e suas relações com o metamodelo BPMN. Sua especificação é feita em UML e ignora qualquer restrição imposta pelos mecanismos de extensão BPMN. Dentro do CDME, os conceitos são inicialmente caracterizados como BPMN *Concepts* ou *Extension Concepts*. BPMN *Concepts* são os elementos nativos presentes no metamodelo BPMN, enquanto que *Extension Concepts* são os elementos definidos no domínio da extensão criada.

Um exemplo apresentado por Stroppi, Chiotti e Villarreal (2011) é de uma extensão do elemento “*UserTask*”, o qual foi estendido com o objetivo de prover uma melhor estratégia de distribuição do trabalho. Esta estratégia provê uma especificação mais detalhada da forma como o trabalho é executado dentro do processo de negócio. A extensão propõe a criação de três novas propriedades: “*distributionAgent*”, indica se a atividade é distribuída pelo sistema ou pelo administrador do processo; “*distributionTrigger*”, especifica o evento causador do início

do processo de distribuição; “*taskPrivileges*”, detalha os privilégios dados aos recursos para completar as atividades. A Figura 17 apresenta três alternativas de CMDE para o exemplo, as quais apresentam BPMN Concepts em cinza e Extension Concepts em branco.

Figura 17: Alternativas CDME.

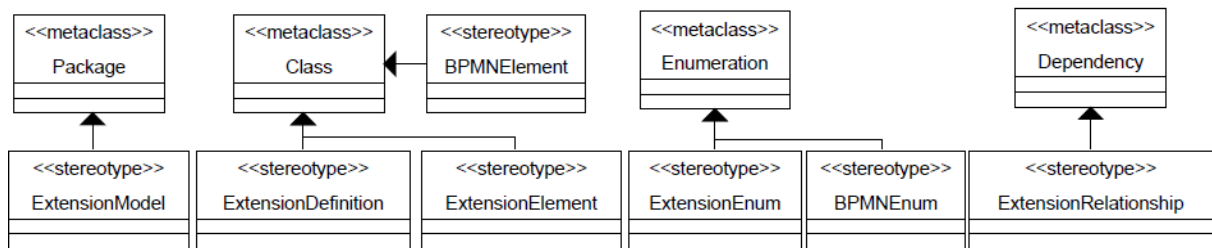


Fonte: Stroppi, Chiotti e Villarreal (2011).

A segunda etapa desta metodologia prevê o desenvolvimento do BPMN *Plus Extension* (BPMN+X), baseado no modelo CDME da etapa anterior, e representado por *profiles* UML. Para os autores, existem dois motivos para o uso de *profiles* UML, são eles: o suporte de diversas ferramentas existentes para construção dos modelos e a efetiva curva de aprendizagem devido a UML ser uma linguagem de modelagem popular no mercado.

Todas as semânticas dos elementos presentes no BPMN+X estão alinhadas com o mecanismo de extensão do metamodelo BPMN 2.0. Para tanto, foi necessário definir alguns estereótipos: “*ExtensionModel*”, “*ExtensionDefinition*”, “*ExtensionElement*”, “*ExtensionEnum*”, “*BPMNEnum*” e “*ExtensionRelationship*”. A Figura 18 ilustra a relação dos estereótipos do BPMN+X com as metaclasses do UML que eles especializam. “*ExtensionModel*” é o nome do grupo de todos os atributos que serão adicionados na extensão. “*BPMNElement*” representa o elemento original do metamodelo BPMN. “*BPMNEnum*” e “*ExtensionEnum*” representam, respectivamente, o conjunto de literais existentes no metamodelo BPMN e no modelo da extensão. “*ExtensionElement*” representa o novo elemento do modelo da extensão, o qual não está disponível no metamodelo BPMN. “*ExtensionDefinition*” corresponde ao nome do grupo de atributos que estão sendo adicionados ao metamodelo BPMN.

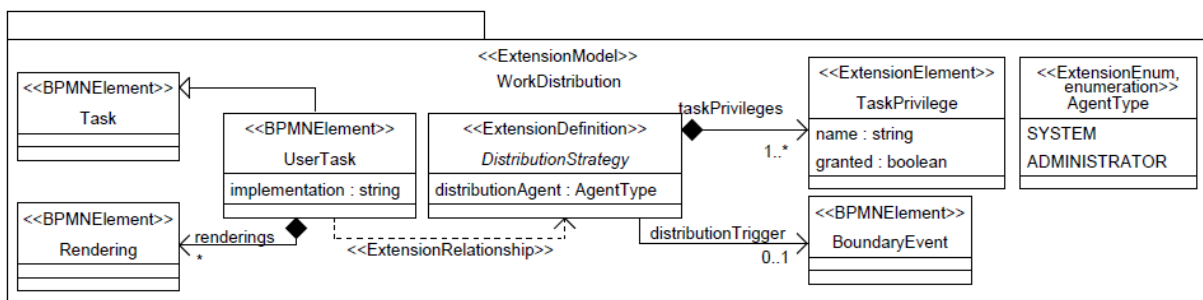
Figura 18: BPMN+X.



Fonte: Stroppi, Chiotti e Villarreal (2011).

O elemento “*ExtensionAttributeDefinition*” do metamodelo BPMN é capturado através das propriedades das classes UML do modelo BPMN+X, sejam elas propriedades do próprio “*ExtensionDefinition*” ou de suas associações. As propriedades dos elementos “*ExtensionDefinition*” e “*ExtensionElement*” podem ser do tipo “*BPMNElement*”, “*ExtensionElement*”, “*ExtensionEnum*”, “*BPMNEnum*” ou mesmo tipos primitivos. O elemento “*ExtensionRelationship*” é utilizado para ligar conceitualmente um “*BPMNElement*” e um “*ExtensionDefinition*”, ou seja, seu uso representa a extensão de um “*BPMNElement*” através de um determinado “*ExtensionDefinition*” ligados pelo “*ExtensionRelationship*” (geralmente utilizados para representar a customização de determinado elemento BPMN). A Figura 19 é um exemplo do modelo BPMN+X resultante do modelo CDME apresentado na Figura 17 após a aplicação de um conjunto de regras de transformações.

Figura 19: Exemplo de um modelo BPMN+X.



Fonte: Stroppi, Chiotti e Villarreal (2011).

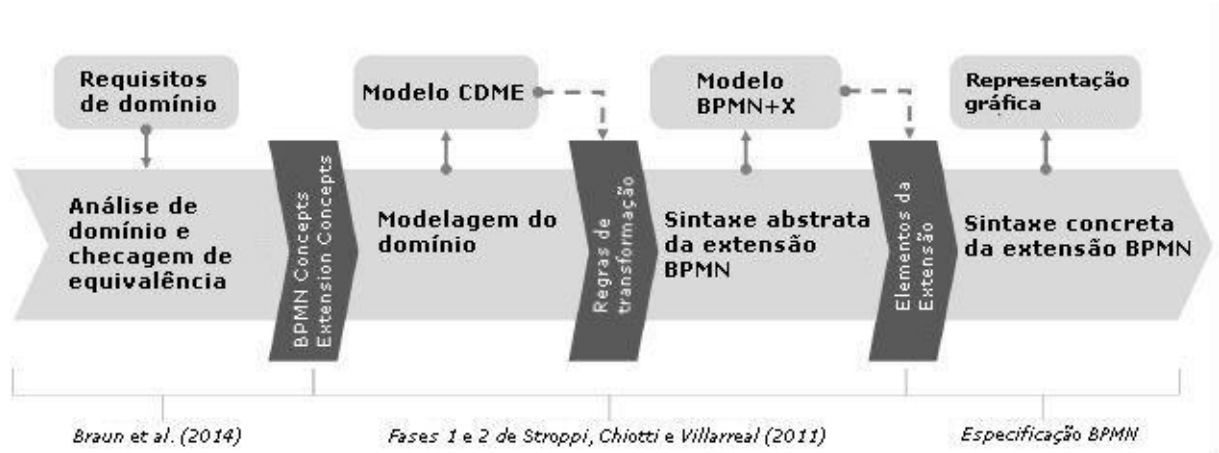
Conforme visto anteriormente, a metodologia de Stroppi, Chiotti e Villarreal (2011) é iniciada ao criar um modelo conceitual do domínio (CDME), o qual destaca os elementos do padrão BPMN e os elementos da extensão proposta. Porém, Braun *et al.* (2014) perceberam que a metodologia não previa uma análise prévia detalhada dos elementos do domínio para identificar se existiam elementos correlatos na notação BPMN, ou seja, durante a criação da extensão poderiam ser criados componentes com uma mesma semântica de componentes do

padrão BPMN. Dessa forma, os autores propõem a criação da fase “Análise do Domínio e Checagem de Equivalência” (*Domain Analysis and Equivalence Check*), a qual é executada antes da criação do CDME e possui dois passos: todos os conceitos necessários e suas semânticas devem ser explicados de acordo com os requisitos da nova linguagem; e uma checagem de equivalência é conduzida a fim de verificar se os conceitos são semelhantes ou não a algum elemento BPMN. Com o objetivo de realizar esta análise, Braun *et al.* (2014) propõem as seguintes regras:

- Equivalência: existe uma semântica equivalente na notação BPMN. Neste caso, nenhuma extensão é necessária e o conceito é representado por um elemento BPMN no CDME.
- Equivalência condicional: não existe uma semântica que claramente indique um elemento BPMN semelhante. Portanto é necessária uma discussão para verificar, individualmente, se o conceito pode ou não ser representado por um elemento BPMN. No final da discussão o conceito será tratado como equivalente ou não equivalente.
- Sem equivalência: não existe nenhuma equivalência por três razões. Primeira, o conceito está completamente fora da linguagem BPMN existente. Segunda, não existe uma relação entre dois conceitos, portanto essa relação precisa ser criada no CDME. Terceira, não existem os atributos necessários para representação do conceito, logo, as propriedades são atribuídas ao CDME.

A Figura 20 ilustra todas as fases seguidas neste trabalho para construção da extensão. Conforme pôde ser observado, foram utilizadas as etapas 1 e 2 da metodologia de Stroppi, Chiotti e Villarreal (2011) e o complemento proposto Braun *et al.* (2014). O processo é iniciado com a análise do domínio e checagem de equivalência, o qual utiliza os requisitos do domínio para sua construção. Em seguida, através da modelagem do domínio, gera-se o CDME que, através da aplicação das regras de transformação, obtém-se o BPMN+X. Finalmente, são gerados os componentes gráficos da extensão proposta.

Figura 20: Fases da metodologia para criação da extensão.



Na próxima seção serão expostos os novos elementos pertencentes ao B4S.ex, apresentando as justificativas de suas criações, suas semânticas e, finalmente, a checagem de equivalência com elementos presentes na notação BPMN.

5.2.2 Análise do domínio e checagem de equivalência

Conforme detalhado na seção anterior, esta etapa da metodologia tem por objetivo analisar os conceitos do domínio problema e compará-los com conceitos existentes na notação BPMN a fim de justificar a criação de uma extensão do elemento ou propriedade. Dessa forma, o Quadro 11 apresenta o mapeamento dos conceitos necessários para mapear serviços através do BPMN, bem como a descrição da semântica dos elementos, classificação de sua equivalência e representação no modelo CDME. Foram destacados detalhes que serão utilizados para facilitar a transformação do BPMN em modelos SoaML.

Os conceitos de Provedor e Consumidor foram identificados como equivalência condicional. O elemento BPMN “*Service Tasks*” é usado para representar uma tarefa (atômica) executada por algum tipo de serviço. Entretanto seu uso não distingue um consumidor de um provedor, característica importante no processo de transformação. Também é necessário adicionar restrições aos objetos de fluxos que podem se conectar a estes elementos, os quais só podem ser interligados através de Invocações.

Quadro 11: Checagem de Equivalência dos novos componentes para o CDME.

Conceito	Semântica	Verificação de Equivalência	CDME
Consumidor	Descreve o conceito que envolve o processo de consumir um serviço. Consumidores iniciam a interação	Equivalência condicional (<i>Service Task</i>).	<i>Extension Concept</i> (especialização de <i>Service Task</i>).

Conceito	Semântica	Verificação de Equivalência	CDME
	com o serviço e recebem o resultado desta interação.		
Provedor	Descreve o conceito que envolve o processo de prover um serviço (seus protocolos, suas especificações e exigências). Responsável pela entrega de valor e resultados das interações entre serviços.	Equivalência condicional (<i>Service Task</i>).	<i>Extension Concept</i> (especialização de <i>Service Task</i>).
Barramento de Serviços	Participante no processo que representa um agrupamento de serviços providos de diferentes formas (sistemas, <i>Enterprise Services Bus</i> , etc).	Equivalência condicional (<i>Pool</i>).	<i>Extension Concept</i> (especialização de <i>Pool</i> ou <i>Participant</i>).
Invocação Simples	Chamado de um consumidor ao provedor do serviço. A chamada é simples, ou seja, não é necessário que o provedor envie dados de retorno ao consumidor.	Equivalência condicional (<i>Message Flow</i>).	<i>Extension Concept</i> (especialização de <i>Message Flow</i>).
Invocação com Retorno	Chamado de um consumidor ao provedor do serviço. A chamada é simples, ou seja, o provedor enviará dados de retorno ao consumidor.	Equivalência condicional (<i>Message Flow</i>).	<i>Extension Concept</i> (especialização de <i>Message Flow</i>).
Operação	Define as operações dentro de um serviço.	Sem equivalência.	<i>Extension Concept</i> .
Parâmetros Primitivos	Define os parâmetros de tipos primitivos dentro de uma operação.	Sem equivalência.	<i>Extension Concept..</i>
Parâmetros Complexos	Define os parâmetros de tipos complexos (Entidades) dentro de uma operação.	Sem equivalência.	<i>Extension Concept</i> .
Entidades	Representa o conjunto de dados envolvidos em uma transação de serviços.	Equivalência condicional (<i>Data Object</i>).	<i>Extension Concept</i> (especialização de <i>Data Object</i>).
Atributos	Define as informações presentes em uma Entidade. São utilizados apenas tipos primitivos.	Sem equivalência.	<i>Extension Concept</i> .

Fonte: Autores (2016).

O conceito de “Barramento de Serviços” também foi caracterizado como equivalência condicional, visto que o elemento “*Pool*” é utilizado para representar um participante e seu papel dentro da execução de um processo. Entretanto, existem regras para mapear elementos dentro de uma “*Pool*” que são divergentes das regras usadas para representar um serviço.

Primeiramente, um serviço é chamado através da demanda de um consumidor, assim como seu término é a entrega da demanda através do serviço, logo, não existe evento de início ou término dentro da “*Pool*”. Outra questão são os objetos de fluxo entre elementos dentro de uma “*Pool*”, os quais não são obrigatórios pelo mesmo motivo apresentado anteriormente (execução por demanda). Esta representação também fere a boa prática em não conectar mais de um objeto de fluxo diretamente ao elemento “Atividade”. Por fim, as ligações entre participantes externos e internos ao “Barramento” devem, obrigatoriamente, ser realizadas por meio de “Invocações” (simples ou com retorno).

A criação da extensão “Invocação” (simples ou com retorno) justifica-se pela necessidade de informações adicionais aos objetos de fluxo, bem como a restrição de uso para apenas os elementos “Consumidor” e “Provedor”. O elemento BPMN “*Message Flow*”, apesar de ter uma semântica semelhante, é utilizado apenas para simples trocas de informações entre diferentes participantes (*Pools*).

Os elementos “*Data Objects*” representam os dados utilizados nas atividades de forma genérica, sem nenhum tipo de detalhamento de suas informações. Entretanto, para realizar a transformação do modelo BPMN e um diagrama de mensagens SoaML é necessário detalhar o tipo de dado utilizado em um serviço, pois existe a possibilidade de utilizá-los como parâmetros de operações. Sendo assim, criou-se o elemento “Entidade”, o qual é restrito a ligar-se apenas a elementos do tipo “Provedor” ou “Consumidor”.

Todos os demais elementos utilizados B4S.ex e não abordados no Quadro 11 foram tipificados como “Equivalentes”, pois os conceitos necessários para representá-los eram semanticamente idênticos aos conceitos dos elementos originais BPMN.

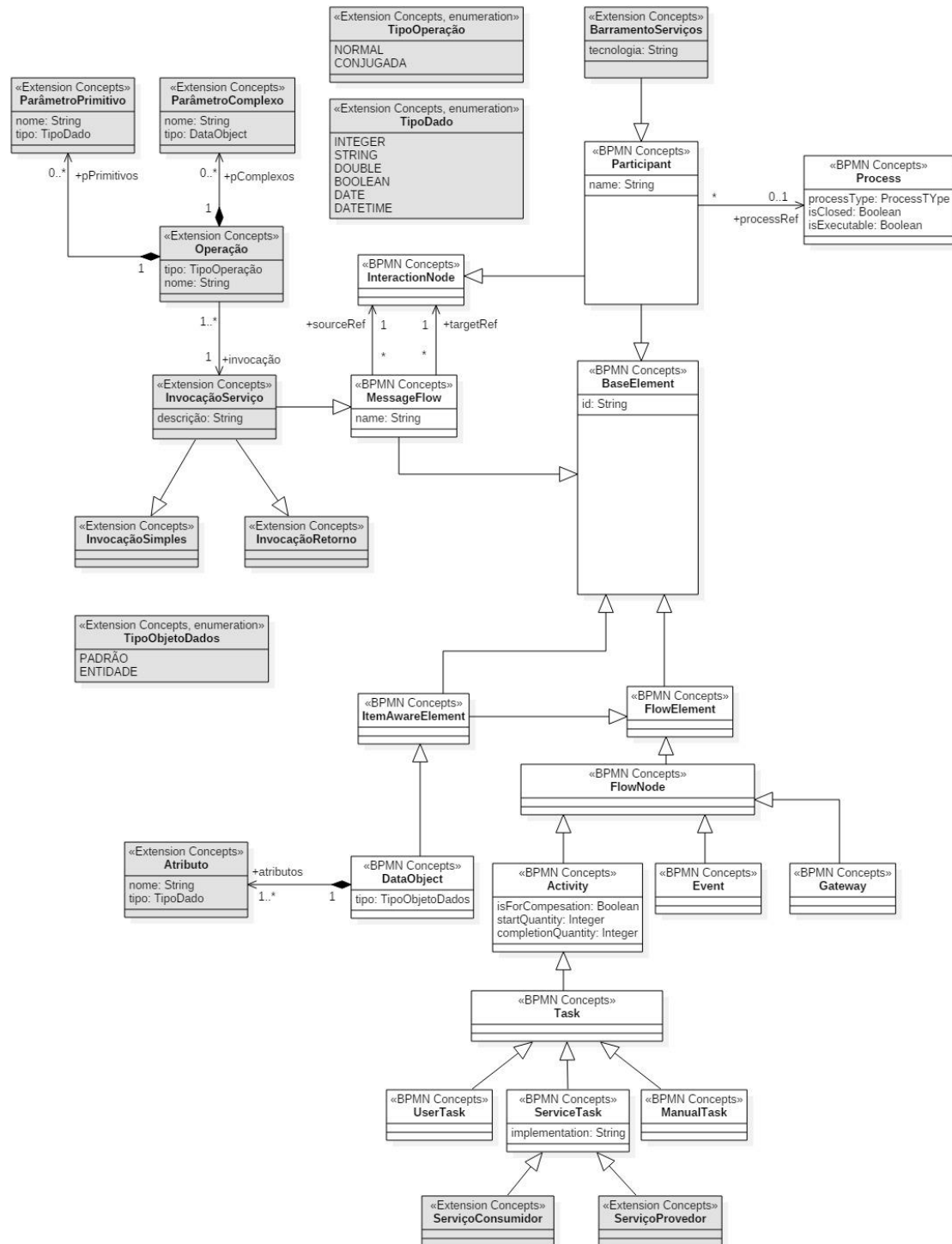
A próxima seção apresentará o modelo CDME, construído com base na análise do domínio e checagem de equivalência, e que será transformado no modelo BPMN+X com base na aplicação de regras de transformações propostas pela metodologia de Stroppi, Chiotti e Villarreal (2011).

5.2.3 Modelo CDME

A segunda etapa da metodologia proposta consiste na criação do *Conceptual Domain Model of the Extension* (CDME), o qual descreve os conceitos do domínio representados pela extensão criada e suas relações com o metamodelo BPMN. Para especificar o modelo CDME

foi utilizado o diagrama de classes da UML através da ferramenta StarUML³, conforme ilustrado na Figura 21.

Figura 21: Modelo CDME para extensão B4S.ex.



Fonte: Autores (2016).

³ Ferramenta para criação de diagramas UML 2.0 mantido pela MKLab desde 2014. Disponível em <<http://staruml.io/>>.

Na Figura 21, os elementos da extensão foram marcados em cinza para uma melhor visualização no modelo e contam com o estereótipo “*Extension Concepts*”, enquanto que os elementos pertencentes ao metamodelo BPMN permaneceram em branco e possuem o estereótipo “*BPMN Concepts*”. O elemento “BarramentoServiços” é a especialização do elemento BPMN “*Participant*” (*Pool*), herdando todas suas características. “InvocaçãoServiços” especializa a classe “*MessageFlow*” do metamodelo BPMN a fim de especificar uma troca de mensagens com uma semântica mais restrita entre elementos, neste caso, entre serviços. “InvocaçãoServiços” ainda conta com suas próprias especializações: “InvocaçãoSimples”, quando a chamada ao serviço não aguarda uma resposta do “Provedor”, e “InvocaçãoRetorno”, quando o “Consumidor” aguarda uma resposta do “Provedor” após sua execução. “Operações” representam todos os métodos que um serviço oferece, desde que passados os parâmetros (“ParâmetroPrimitivo” ou “ParâmetroComplexo”) corretos. “TipoOperação” reflete se determinada operação será “Normal” ou “Conjugada”, fator importante na representação em SoaML após as transformações. Os parâmetros de uma operação, quando sendo do tipo “Complexo”, podem ser representados por um elemento da extensão que especializa a classe “*DataObject*”, o “EntidadeServiço”, o qual conta com um conjunto de atributos (classe “*Atributo*”). Por fim, foram criados os elementos “ServiçoConsumidor” e “ServiçoProvedor” especializando o elemento BPMN “*ServiceTask*”.

5.2.4 Modelo BPMN+X

Seguindo a metodologia, o segundo passo consiste na criação do modelo BPMN+X com base no CDME construído na primeira etapa. Para o desenvolvimento do BPMN+X, a metodologia apresenta um procedimento com uma série de etapas e regras de transformação que devem ser seguidas.

Primeiramente, deve-se transformar todos os elementos do CDME caracterizados com o “*BPMN Concept*” em “*BPMNElement*” ou, quando caracterizado como “*Enumeration*”, “*BPMNEnum*”. Em seguida transforma-se em “*ExtensionEnum*” todas as classes “*Enumeration*” do tipo “*Extension Concept*”.

Seguindo o procedimento, aplicam-se as regras de transformação apresentadas no Quadro 12, as quais são aplicadas às propriedades do modelo CMDE, resultando em suas representações no modelo BPMN+X. Para tanto, entende-se c uma classe do CDME, onde p é uma propriedade do tipo t que é um atributo de c ou é navegável de c através da associação para t . Existem alguns valores que podem ser atribuídos a estas variáveis para definir as representações de c , p e t no modelo BPMN+X, são elas:

- Se c é identificado como um “BPMN Concept” ou “Extension Concept”.
- Se p é uma propriedade nova ou original de c .
- Se t é um “BPMN Concept”, “Extension Concept” ou “Data Type”.

Uma propriedade p é considerada original quando existe uma propriedade p na classe c no metamodelo BPMN. Caso contrário, p é considerada uma nova propriedade. Por fim, t é um “Data Type” quando pertence a alguma classe “Enumeration” ou é composta de tipos primitivos. O Quadro 12 resume estas características.

Quadro 12: Regras para representar as propriedades das classes do CDME no modelo BPMN+X.

Classe (c)	Propriedade (p)	Tipo (p)	Representação de p
1a BPMN Concept	Original	Data Type	BPMNElement Property
1b BPMN Concept	Original	BPMN Concept	BPMNElement Property
2a BPMN Concept	Nova	Data Type	ExtensionAttributeDefinition
2b BPMN Concept	Nova	BPMN Concept	ExtensionAttributeDefinition
2c BPMN Concept	Nova	Extension Concept	ExtensionAttributeDefinition
3 BPMN Concept	Nova	Extension Concept	ExtensionRelationship
4a Extension Concept	Nova	Data Type	ExtensionElement Property / ExtensionAttributeDefinition
4b Extension Concept	Nova	BPMN Concept	ExtensionElement Property / ExtensionAttributeDefinition
4c Extension Concept	Nova	Extension Concept	ExtensionElement Property / ExtensionAttributeDefinition

Fonte: Adaptado de Stroppi, Chiotti e Villarreal (2011).

A regra 1 é aplicada para todas as propriedades originais de elementos “BPMN Concept” do CDME. Se t é um “Data Type”, então p é representado no BPMN+X como um atributo de “BPMNElement” equivalente a c (regra 1a). Caso t seja um “BPMN Concept”, p é representado como uma associação do “BPMNElement” equivalente a c para o “BPMNElement” equivalente a t no BPMN+X. No modelo CDME criado (Figura 21), a regra 1a foi aplicada nas propriedades “name” (Participant), “name” (MessageFlow), “iscloded” (Process), “isExecutable” (Process), “isForCompensation” (Activity), “startQuantity” (Activity), “completionQuantity” (Activity) e “implementation” (ServiceTask), visto que todas são propriedades de “BPMN Concepts” com propriedades originais e tipos primitivos (“Data Type”). As propriedades transformadas pela regra 1b do modelo CDME criado foram: “sourceRef” (MessageFlow), “targetRef” (MessageFlow) e “processRef” (Participant).

De acordo com a regra 2, se p é uma nova propriedade de um “BPMN Concept”, então p é especificado no modelo BPMN+X como um “*ExtensionAttributeDefinition*” definido como uma propriedade de um elemento “*ExtensionDefinition*”. Ou seja, um “*ExtensionDefinition*” deve ser relacionado com o “*BPMNElement*” equivalente a c através de um “*ExtensionRelationship*”. Em seguida, caso t seja um “*Data Type*” (regra 2a), p é especificado como um atributo no elemento “*ExtensionElement*”. Caso t seja um “BPMN Concept”, então p é especificado como uma associação do elemento “*ExtensionDefinition*” criado para o “*BPMNElement*” equivalente t no modelo BPMN+X. A regra 2c é aplicada quando t é um “*Extension Concept*”, ou seja, um novo tipo de elemento que foi criado na extensão proposta. Neste caso, p é um “*ExtensionAttributeDefinition*” representado como uma associação do elemento “*ExtensionDefinition*” criado para o “*ExtensionElement*” equivalente a t no modelo BPMN+X. Neste trabalho, foi aplicada apenas a regra 2c, pois o elemento “*DataObject*” possui a propriedade “atributos” do tipo “Atributo” (“*Extension Concept*”). Por tanto, no modelo BPMN+X, criou-se um “*ExtensionDefinition*” nomeado “EntidadeServiço”, conectado através de um “*ExtensionRelationship*” a “*DataObject*”, e ligado ao “*ExtensionElement*” “Atributo”.

Aplica-se a regra 3 quando p é uma nova propriedade de um “BPMN Concept” e t é “*Extension Concept*” abstrato utilizado para agrupar um conjunto de propriedades adicionadas a c . Neste caso, representa-se t como uma “*ExtensionDefinition*” e p como um “*ExtensionRelationship*” do “*BPMNElement*” equivalente a c para o “*ExtensionDefinition*” criado. A regra 3 não foi aplicada nesta extensão.

A regra 4 ocorre quando p é uma propriedade de um “*Extension Concept*”. Se t é um “*Data Type*”, então p é especificado como um atributo do elemento equivalente a c (regra 4a). Caso t seja um “BPMN Concept”, p é caracterizado como uma associação entre o elemento equivalente a c para o elemento equivalente a t (regra 4b). Caso t seja um “*Extension Concept*”, t é especificado por um “*ExtensionElement*” e p por uma associação do elemento equivalente a c para o elemento equivalente a t (regra 4c). Nesta extensão, a regra 4a foi aplicada nas seguintes propriedades: “nome” (“ParâmetroPrimitivo”), “tipo” (“ParâmetroPrimitivo”), “nome” (“ParâmetroComplexo”), “tipo” (“ParâmetroComplexo”), “tipo” (“Operação”), “nome” (“Operação”), “descrição” (“InvocaçãoServiço”), “nome” (“Atributo”) e “tipo” (“Atributo”). Enquanto que para a regra 4c, as seguintes propriedades foram afetadas: “atributos” (“*DataObject*”), “invocação” (“Operação”), “pPrimitivos” (“Operação”) e “pComplexos” (“Operação”). O Quadro 13 resume o resultado da aplicação das regras com base no CDME criado.

Quadro 13: Regras aplicadas às propriedades do CDME.

Regra	Elemento (<i>c</i>)	Propriedade (<i>p</i>)
1a	<i>Participant</i>	<i>name</i>
	<i>MessageFlow</i>	<i>name</i>
	<i>Process</i>	<i>isClosed</i> <i>isExecutable</i>
	<i>Activity</i>	<i>isForCompensation</i> <i>startQuantity</i> <i>completionQuantity</i>
	<i>ServiceTask</i>	<i>Implementation</i>
1b	<i>MessageFlow</i>	<i>sourceRef</i> <i>targetRef</i>
	<i>Participant</i>	<i>processRef</i>
2c	Atributo	atributos
4a	ParâmetroPrimitivo	nome tipo
	ParâmetroComplexo	nome tipo
	Operação	nome tipo
	InvocaçãoServiço	descrição
	Atributo	nome tipo
4c	DataObject	Atributos
	Operação	invocação pPrimitivos pComplexos

Fonte: Autores (2016).

Além de analisar as propriedades, é necessário verificar as associações de generalizações dentro do modelo CMDE. Para tanto, seja *g* um relacionamento de generalização de uma classe *c* para uma superclasse *s*, a representação de *g*, *c* e *s* depende da caracterização de *c* e *s* e se *g* é um relacionamento de generalização novo ou original. As regras de generalizações estão resumidas no Quadro 14.

Quadro 14: Regras para representar os relacionamentos de generalizações do CDME no modelo BPMN+X.

Classe (<i>c</i>)	Generalização (<i>g</i>)	Superclasse (<i>s</i>)	Representação de <i>g</i>
5 BPMN Concept	Original	BPMN Concept	BPMNElement Generalization
6 BPMN Concept	Novo	BPMN Concept	Inválido
7a BPMN Concept	Novo	Extension Concept	ExtensionRelationship
7b Extension Concept	Novo	BPMN Concept	ExtensionRelationship
8a Extension Concept	Novo	Extension Concept	ExtensionElement Generalization
8b Extension Concept	Novo	Extension Concept	ExtensionElement Generalization

Fonte: Traduzido de Stroppi, Chiotti e Villarreal (2011).

Seguindo a análise, a regra 5 é utilizada quando *g* é uma generalização original de um “BPMN Concept” *c* para outro “BPMN Concept” *s*. Dessa forma, *g* é representado como um relacionamento de generalização do “BPMNElement” equivalente a *c* para o “BPMNElement” equivalente a *s* no modelo BPMN+X. A regra 5 é aplicada no modelo CDME nos seguintes relacionamentos de generalização: “Participant” e “BaseElement”, “Participant” e “InteractionNode”, “BaseElement” e “FlowElement”, “FlowElement” e “FlowNode”, “FlowNode” e “Activity”, “FlowNode” e “Event”, “FlowNode” e “Gateway”, “DataObject” e “ItemAwareElement”, “FlowElement” e “ItemAwareElement”, “Task” e “User Task”, “Task” e “ServiceTask”, “Task” e “ManualTask”.

A regra 6 é aplicada quando *g* é um novo relacionamento de generalização de um “BPMN Concept” para outro “BPMN Concept”. Contudo, neste caso, a regra é inválida, pois o mecanismo de extensão BPMN 2.0 não permite criação de novas estruturas entre elementos do seu metamodelo original.

A condição da regra 7 é *g* ser uma relação de generalização entre um “BPMN Concept” e um “Extension Concept”. O mecanismo disponibilizado pelo BPMN só permite criação de extensão por adição, ou seja, não é permitido representar uma nova relação taxonômica entre um elemento mais específico “BPMN Concept” e um mais geral “Extension Concept” (regra 7a), ou vice-versa (regra 7b), pois *g* seria inválido. Entretanto, para solucionar este problema, os autores propõem a representação do “Extension Concept” como um “ExtensionDefinition” e *g* como um “ExtensionRelationship” do elemento “BPMNElement” (equivalente ao “BPMN Concept” do modelo CDME) ligado ao “ExtensionDefinition” criado. Dessa forma, é possível

adicionar as propriedades definidas no “*Extension Concept*” no elemento “BPMN Concept” correspondente. A regra 7b foi aplicada nos elementos “ServiçoConsumidor”, “ServiçoProvedor”, “BarramentoServiços” e “InvocaçãoServiços”

Por último, tem-se a regra 8, a qual é aplicável quando *g* é uma generalização de um “*Extension Concept*” para outro “*Extension Concept*” (é necessário que *s* seja previamente classificado como um “*ExtensionDefinition*” ou “*ExtensionElement*” através de alguma das regras anteriores). Dessa forma, *c* é representado como um “*ExtensionElement*” (regra 8a) ou como “*ExtensionDefinition*” (regra 8b) dependendo se *s* é um “*ExtensionElement*” ou “*ExtensionDefinition*”, respectivamente. A representação de *g* é feita com um relacionamento de generalização do elemento equivalente a *c* para o elemento equivalente a *s* no modelo BPMN+X. A regra 8b foi aplicada nos elementos “InvocaçãoSimples” e “InvocaçãoRetorno” do modelo CDME.

O Quadro 15 resume as regras aplicadas no modelo CDME para transformação das associações de generalização da extensão proposta no modelo BPMN+X.

Quadro 15: Regras aplicadas aos relacionamentos de generalização do CDME.

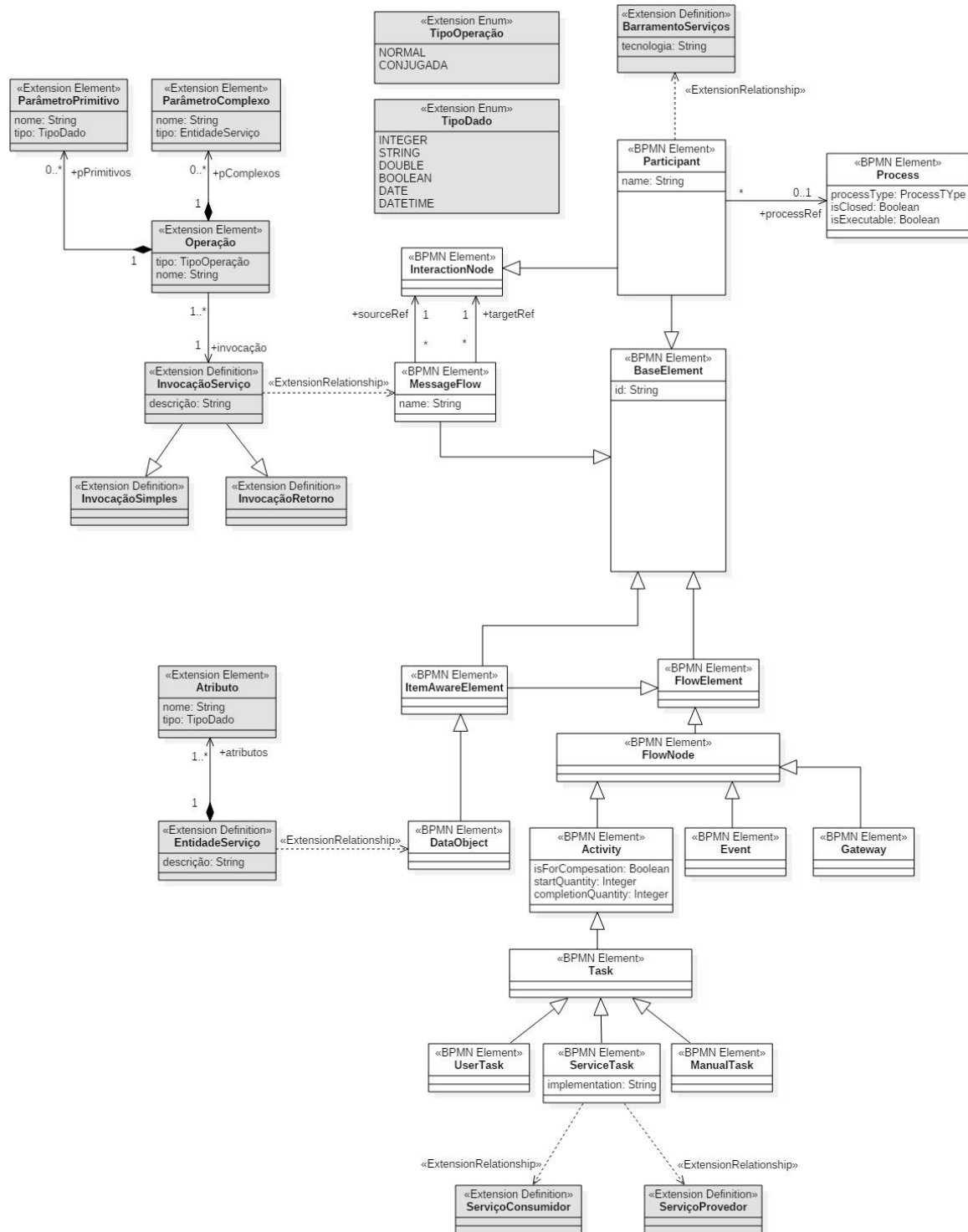
Regra	Classe (c)	Super Classe (s)
5	<i>Participant</i>	<i>BaseElement</i>
	<i>Participant</i>	<i>InteractionNode</i>
	<i>FlowElement</i>	<i>BaseElement</i>
	<i>FlowNode</i>	<i>FlowElement</i>
	<i>Activity</i>	<i>FlowNode</i>
	<i>Event</i>	<i>FlowNode</i>
	<i>Gateway</i>	<i>FlowNode</i>
	<i>DataObject</i>	<i>ItemAwareElement</i>
	<i>ItemAwareElement</i>	<i>FlowElement</i>
	<i>Task</i>	<i>UserTask</i>
	<i>Task</i>	<i>ServiceTask</i>
	<i>Task</i>	<i>ManualTask</i>
7b	ServiçoConsumidor	<i>ServiceTask</i>
	ServiçoProvedor	<i>ServiceTask</i>
	BarramentoServiço	<i>Participant</i>
	InvocaçãoServiço	<i>MessageFlow</i>
8b	InvocaçãoSimples	InvocaçãoServiço
	InvocaçãoRetorno	InvocaçãoServiço

Fonte: Autores (2016).

Após a aplicação de todas as regras de transformações propostas por Stroppi, Chiotti e Villarreal (2011), uma representação do metamodelo da extensão foi alcançada e representada através do modelo BPMN+X da metodologia. O metamodelo da extensão criada, o qual tem

por objetivo auxiliar as transformações do *framework* apresentado neste trabalho, pode ser visto na Figura 22.

Figura 22: Modelo BPMN+X.



Fonte: Autores (2016).

Em seguida serão ilustrados todos os componentes gráficos propostos a partir da criação do B4S.ex para integrar o metamodelo BPMN.

5.2.5 Novas propriedades e elementos gráficos da extensão

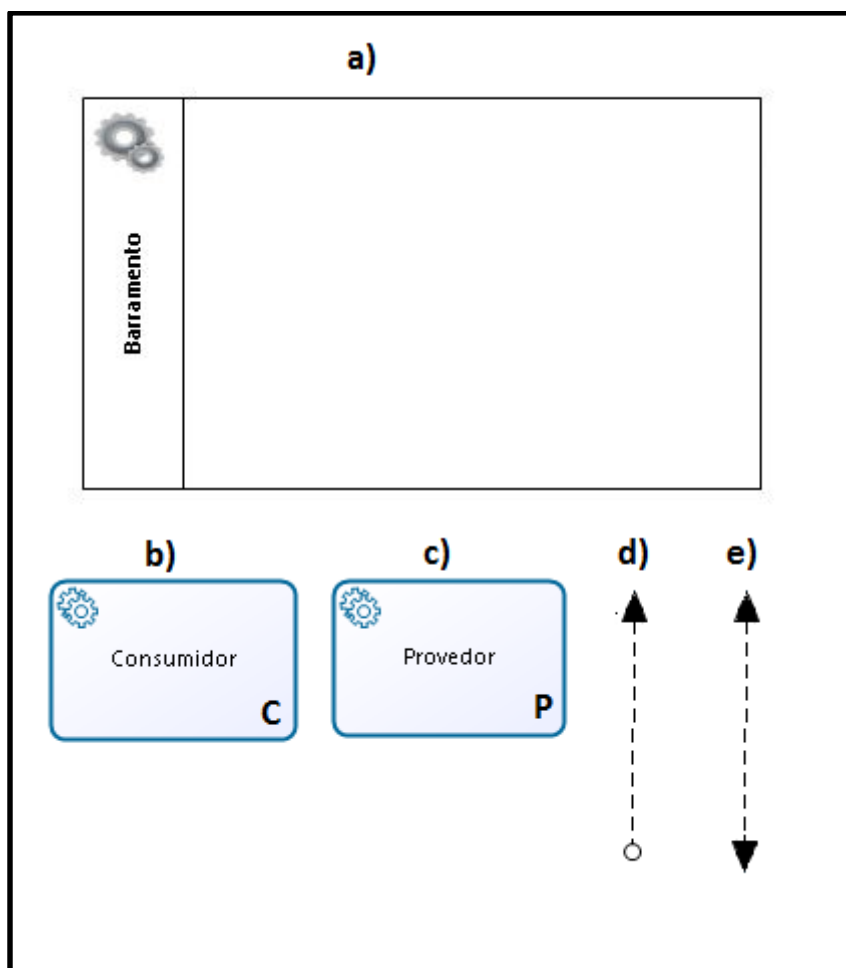
Conforme notado ao longo das atividades realizadas anteriormente, a conformidade com o padrão BPMN é uma das premissas para criação desta extensão. Dessa forma, além da conformidade do metamodelo estendido com o metamodelo original BPMN, utilizando seu próprio mecanismo de extensão, também foi preservado os requisitos para criação de novos elementos gráficos dentro da notação.

De acordo com a OMG (2011), o elemento chave do BPMN é a definição das formas e ícones usados para representar seus elementos gráficos na especificação, os quais serão compreendidos por todos os profissionais envolvidos. Ou seja, diagramas BPMN de processos de negócio devem utilizar os elementos, formas e marcadores ilustrados em sua especificação. Contudo, existe uma série de características flexíveis presentes na especificação BPMN que permitem a criação de novas estruturas ou componentes, são elas:

- Novos marcadores ou indicadores podem ser adicionados para especificar elementos gráficos. Estes elementos podem ser usados para destacar um atributo específico de um elemento BPMN ou representar novos subtipos de conceitos correlatos.
- Uma nova forma representando um novo tipo de artefato pode ser adicionado a um diagrama, mas a forma deste novo artefato não deve entrar em conflito com formas de nenhum outro elemento ou marcador BPMN.
- Elementos gráficos podem ser coloridos e as cores podem representar uma semântica estendida da especificação do elemento.
- O estilo da linha de um elemento gráfico pode ser modificado, porém não poderá entrar em conflito com nenhum outro estilo de linha requerido pela especificação.
- Uma extensão não deve mudar a forma de um elemento gráfico definido na especificação do padrão. Ex.: transformar retângulos em losangos, ou mudar o formato da borda arredondada para reta, etc.

Em resumo, a notação BPMN segue um tipo de extensão denominado “por adição”, ou seja, elementos novos podem ser adicionados, desde que sigam seu padrão e, principalmente, que não entrem em conflitos com nenhum outro elemento presente em sua especificação, seja em sua estrutura, semântica ou mesmo em representação gráfica. Com base nestes requisitos, foram criados os elementos gráficos apresentados na Figura 23.

Figura 23: Componentes adicionados na extensão.



Fonte: Autores (2016).

Na Figura 23, pode-se verificar a criação de cinco novos elementos gráficos: “Barramento de Serviços” (23.a), “Consumidor” (23.b), “Provedor” (23.c), “Invocação Simples” (23.d) e “Invocação com Retorno” (23.e). A necessidade da criação de cada elemento foi apresentada na etapa “Análise de Domínio e Checagem de Equivalência”, bem como as restrições que serão impostas à utilização de cada um.

Além dos novos componentes, também foram adicionadas novas propriedades, tanto nos novos elementos quanto em elementos nativos BPMN. No elemento “*Data Object*” foram inseridas as propriedades para identificar os dados e seus tipos. No elemento “Invocação Simples” é possível identificar o nome da operação e seus parâmetros, enquanto que no elemento “Invocação com Retorno”, além de exibir a operação normal, também é possível identificar a operação conjugada e seus parâmetros.

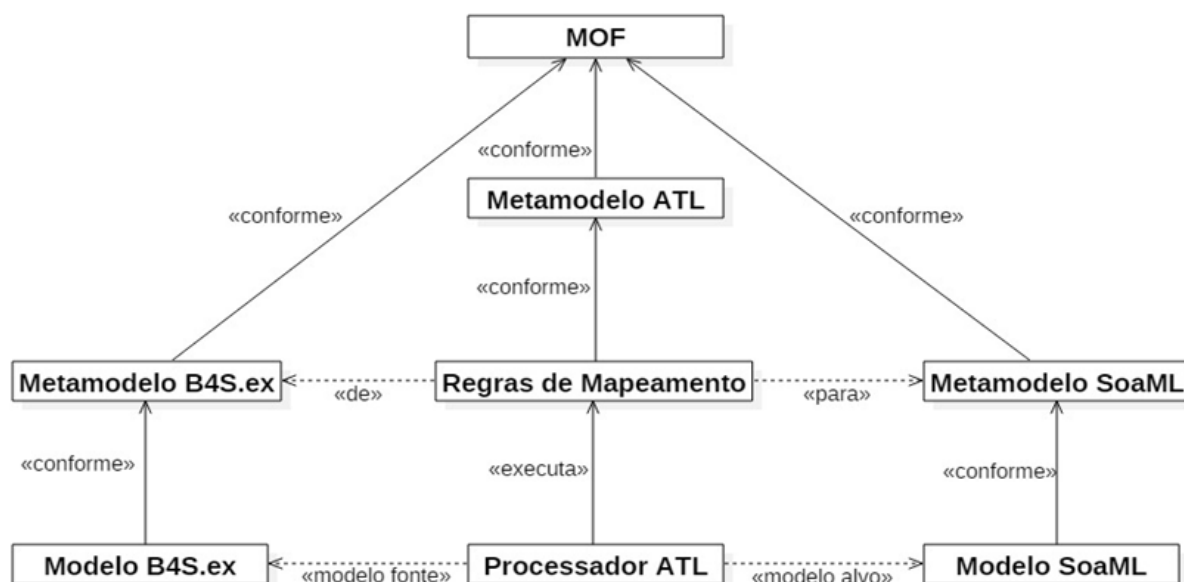
As próximas seções serão responsáveis por apresentar o processo de transformação de um modelo criado através do B4S.ex para o modelo SoaML, detalhando as regras de

mapeamento definidas entre os metamodelos bem como a linguagem de transformação utilizada.

5.3 Transformação: B4S.ex para SoaML

Conforme visto no começo deste capítulo, um dos primeiros processos de transformação que compõe o BPM4Services é a passagem de um modelo construído em B4S.ex para um modelo representado em SoaML. Para conduzir o processo de transformação, o BPM4Services utiliza a estrutura apresentada na Figura 24, auxiliado pelo *Atlas Transformation Framework* (ATL).

Figura 24: Estrutura ATL.



Fonte: Autores (2016).

Na Figura 24, pode-se observar que todos os metamodelos envolvidos neste processo de transformação foram desenvolvidos em conformidade com o padrão MOF. Para implementação da transformação, primeiramente, é necessário criar as regras de mapeamento entre os metamodelos, os quais, neste caso, são o metamodelo B4S.ex e o metamodelo SoaML. As regras de mapeamento definem as relações entre os componentes do modelo fonte (B4S.ex) e do modelo alvo (SoaML). Finalmente, através da execução das regras de mapeamento em um processador ATL, é possível gerar um modelo SoaML destino a partir de um modelo B4S.ex origem. Dessa forma, observa-se que, para um processo de conversão completo, são necessários quatro documentos: o metamodelo fonte, o metamodelo alvo, o modelo fonte e as regras de mapeamento. Após concluído o processo, gera-se o modelo alvo.

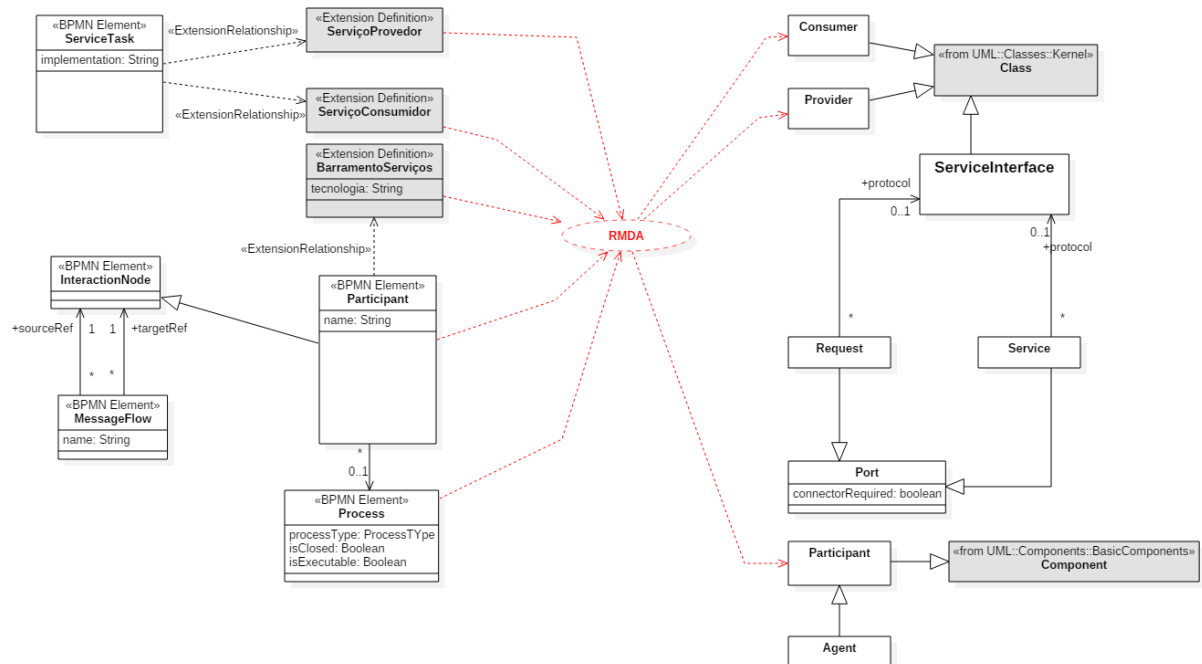
A fim de facilitar o entendimento das relações entre os metamodelos, as regras de mapeamentos foram separadas de acordo com o diagrama que será gerado no modelo SoaML, são elas:

- Regras de Mapeamento para o Diagrama de Arquitetura (RMDA).
- Regras de Mapeamento para o Diagrama de Participantes (RMDP).
- Regras de Mapeamento para o Diagrama de Interfaces (RMDI).
- Regras de Mapeamento para o Diagrama de Mensagens (RMDM).

A Figura 25 apresenta o mapeamento completo entre os metamodelos B4S.ex e SoaML, definindo as regras responsáveis pela transformação dos modelos. Na representação utilizou-se o símbolo da elipse pontilhada para identificar as regras de mapeamentos. O critério utilizado para fazer a ligação entre os componentes dos metamodelos foi a necessidade do objeto origem, e suas propriedades, serem percorridas para execução de uma determinada regra e, conseqüentemente, construção do objeto no modelo alvo. Para facilitar o entendimento, cada regra será explicada separadamente a seguir.

A criação do diagrama de arquitetura é realizada através da regra RMDA, a qual é iniciada realizando uma leitura dos objetos “*Process*”, “*Participant*”, “*BarramentoServiços*”, “*ServiçoConsumidor*” e “*ServiçoProvedor*” do modelo B4S.ex, conforme ilustrada na Figura 26.

Figura 26: Regras de Mapeamento para Diagrama de Arquitetura.

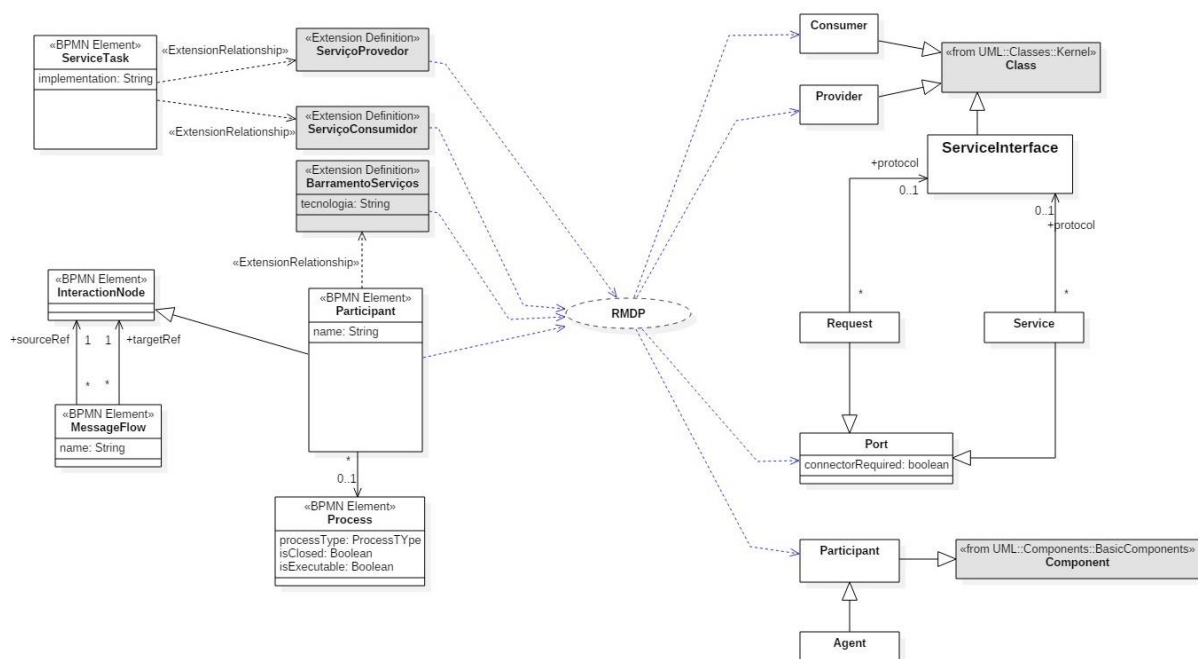


Fonte: Autores (2016).

Primeiramente, os objetos do tipo “*Participant*” no modelo B4S.ex tornam-se do tipo “*Participant*” no modelo SoaML. Em seguida, o objeto dentro do modelo do processo de negócio que representa o barramento de serviços (“BarramentoServiços”) é recuperado e, através dele, são buscadas as referências dos elementos “ServiçoProvedor” a fim de transformá-los em elementos do tipo “*Provider*” no modelo SoaML. Também é realizada uma busca para identificar quais participantes contém elementos “ServiçoConsumidor” pois, dessa forma, são definidas as ligações dentro do diagrama de arquitetura entre os serviços e seus participantes. A propriedade “*name*” do objeto do tipo “*Process*” definirá o nome do diagrama de arquitetura.

A criação do diagrama de participantes é realizada através da regra RMDP, a qual é iniciada realizando uma leitura dos objetos “*Participant*”, “ServiçoConsumidor” e “ServiçoProvedor” do modelo B4S.ex, conforme ilustrada na Figura 27.

Figura 27: Regras de Mapeamento para Diagrama de Participantes.



Fonte: Autores (2016).

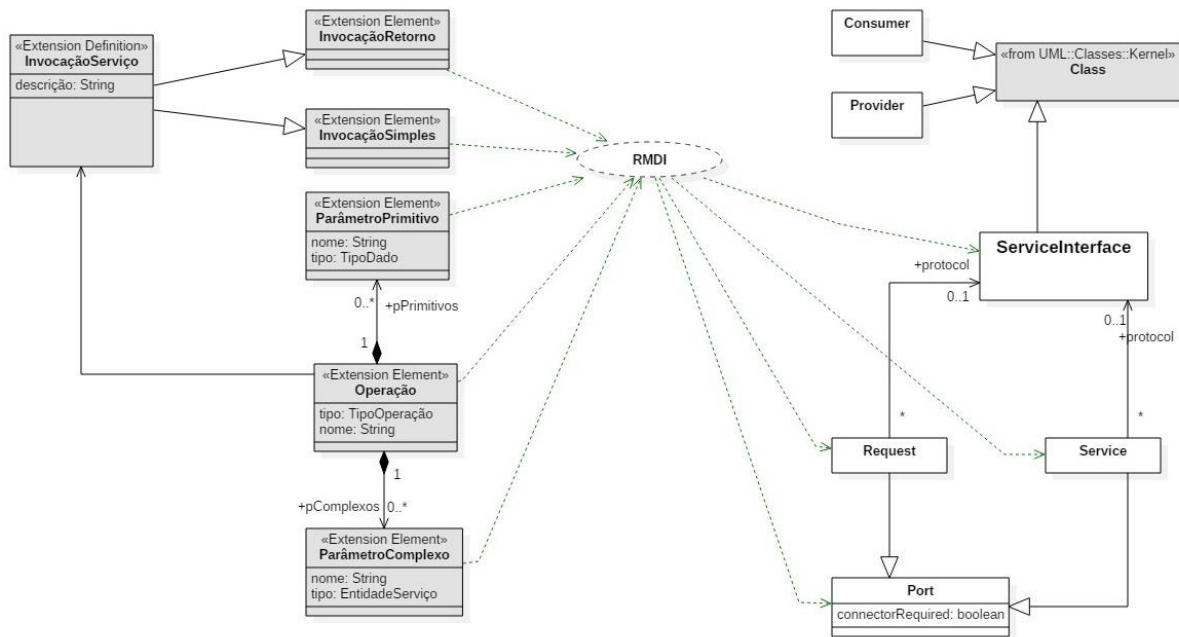
O primeiro passo para criação do diagrama de participantes é recuperar os participantes do modelo origem e transformá-los no elemento “*Participant*” no modelo destino. Em seguida o elemento “BarramentoServiço” é recuperado e transformado em um “*Participant*”, porém o participante originado dessa forma conterá todos os elementos “*Provider*” (originados da leitura dos “ServiçoConsumidor” do modelo B4S.ex). Os demais serviços (“ServiçoConsumidor”) serão colocados em seus respectivos participantes através da transformação em elementos do tipo “*Consumer*”. Para associar serviços e participantes no

modelo destino, faz-se necessário criar elementos do tipo “Port”, os quais são responsáveis por realizar a interação entre os serviços e o ambiente externo ao participante.

O diagrama de interface é responsável por especificar e definir a forma como são relacionados as “*simple interfaces*” (ou interface simples) e “*service interfaces*” (ou interface de serviços). “Interfaces simples” realizam uma interação simples em um caminho único, ou seja, este tipo de interação, chamadas de “anônimas”, não necessita de informações sobre os consumidores nem de protocolos definidos. Por outro lado, “interfaces de serviços” permite uma comunicação bilateral entre consumidor e provedor, ou seja, são especificadas as interfaces necessários para consumir o provedor, bem como as interfaces necessárias para o provedor se comunicar com o consumidor.

Sendo assim, a criação do diagrama de interfaces é realizada através da regra RMDI, a qual é iniciada realizando uma leitura dos objetos “InvocaçãoRetorno”, “InvocaçãoSimples”, “ParâmetroPrimitivo”, “Operação” e “ParâmetroComplexo” do modelo B4S.ex, conforme ilustrada na Figura 28.

Figura 28: Regras de Mapeamento para Diagrama de Interfaces.



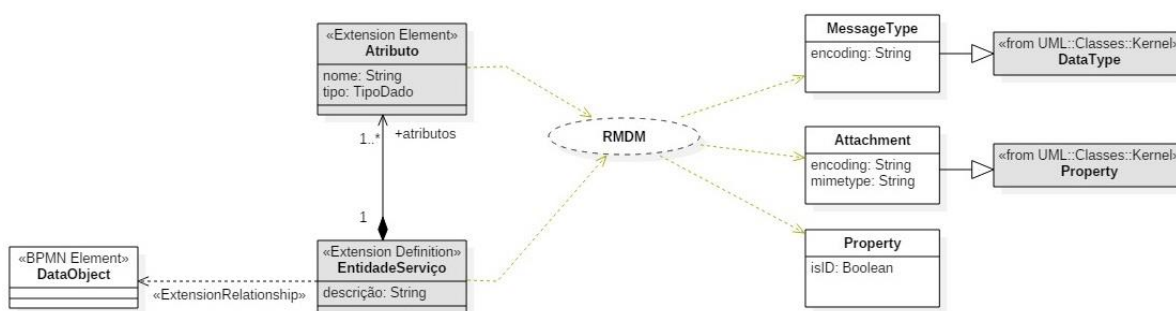
Fonte: Autores (2016).

O elemento “InvocaçãoServiço” possui duas especializações: “InvocaçãoRetorno” e “InvocaçãoSimples”. Ambos os tipos de invocação se transformam em “ServiceInterfaces”, porém as interfaces geradas através dos elementos do tipo “InvocaçãoSimples” apenas realizam as interfaces provedoras, enquanto que os elementos do tipo “InvocaçãoRetorno” realizam as interfaces provedoras e usam as interfaces consumidoras. As invocações possuem operações,

as quais representam os métodos que são criados nas interfaces geradas durante a transformação. Os elementos “ParâmetroComplexo” e “ParâmetrosPrimitivos” irão compor os parâmetros das operações nas interfaces do modelo destino. Sendo assim, a partir da perspectiva do provedor, são identificadas as invocações de cada serviço existente no modelo origem a fim de que se possa classificar as interfaces em consumidoras ou provedoras, inserir as operações e, finalmente, os parâmetros correspondentes.

A criação do diagrama de mensagens é realizada através da regra RMDM, a qual é iniciada realizando uma leitura dos objetos “EntidadeServiço” e “Atributo” do modelo B4S.ex, conforme ilustrada na Figura 29.

Figura 29: Regras de Mapeamento para Diagrama de Mensagens.



Fonte: Autores (2016).

Para obter o diagrama de mensagens, é necessário percorrer basicamente dois tipos de elementos: “EntidadeServiço”, o qual identifica um objeto de dados no processo de negócio, e o elemento “Atributo”, que descreve todos os dados existentes em uma entidade. Dessa forma, para cada “EntidadeServiço” será criado um elemento “*MessageType*”, agrupando todos seus atributos. Para identificar as relações entre os elementos “*MessageType*” do diagrama de mensagens é necessário verificar os tipos dos atributos, quando houver dados que sejam outras entidades, então esses elementos “*MessageType*” serão conectados com uma associação.

5.4 Considerações Finais do Capítulo

Neste capítulo foi descrito o *framework* BPM4Services e sua estrutura, baseada na arquitetura MDA da OMG e nos conceitos de BPM e SOA. O *framework* apresentado propôs a criação de uma extensão do metamodelo BPMN, nomeada de B4S.ex. Dessa forma, a partir da especificação de um modelo utilizando o B4S.ex poderiam ser aplicadas diferentes

transformações que pudessem resultar em uma automação do processo de negócio através da composição de uma arquitetura orientada a serviços.

Conforme mencionado no começo do capítulo, devido ao pouco tempo disponível para o desenvolvimento deste trabalho, optou-se por apresentar todo o *framework* e suas características e aprofundar-se na extensão do metamodelo BPMN e sua transformação inicial em um modelo SoaML. Esta decisão foi pautada no entendimento que estes são os principais modelos dentro do *framework*, pois a partir deles que serão gerados os demais modelos e transformações.

Para ilustrar e avaliar a utilização da extensão proposta neste capítulo, bem como a aplicação das regras de transformações especificadas e o modelo resultante em SoaML, foi desenvolvido um caso exemplo, o qual será apresentado no próximo capítulo.

CAPÍTULO 6

CASO EXEMPLO: TRIAGEM NEONATAL

Este capítulo descreve a utilização do BPM4Services por meio de um caso exemplo com o objetivo de validar a aplicabilidade do *framework*. O objeto de estudo foi o Serviço de Referência de Triagem Neonatal (SRTN) do Hospital Universitário da Universidade Federal de Sergipe (HU-UFS), o qual segue as diretrizes do Programa Nacional de Triagem Neonatal (PNTN). Para tanto, realizou-se o mapeamento do processo da Triagem Neonatal com a extensão B4S.ex proposta pelo BPM4Services e, em seguida, aplicaram-se as regras RMDA, RMDP, RMDI e RMDM para geração dos diagramas SoaML.

6.1 O Hospital Universitário e o Programa de Triagem Neonatal (PNTN)

O HU-UFS é um hospital-escola vinculado à Universidade Federal de Sergipe desde 1984, o qual presta assistência médico-hospitalar de média e alta complexidade e serve de base para as atividades acadêmicas dos cursos de saúde. O HU-UFS também atua em programas de assistência e inclusão social por meio de parcerias com órgãos públicos, bem como desenvolve atividades de natureza preventiva e extensiva através de programas nacionais de saúde e educação oferecidos à população sergipana (HU-UFS, 2012).

O hospital não realiza atendimentos particulares ou através de planos de saúde. Na verdade, o HU-UFS presta serviços para o Sistema Único de Saúde (SUS), sendo referência em atendimentos de média e alta complexidade no ambulatório e em relação a exames complementares, diagnóstico e internação. Atualmente o HU possui 123 leitos, 68 consultórios ambulatoriais, realiza mais de 12 mil consultas ambulatoriais e 200 cirurgias por mês em diversas especialidades.

O Programa Nacional de Triagem Neonatal, da Portaria GM/MS 822/2001, tem por objetivo identificar distúrbios de doenças no recém-nascido a fim realizar uma intervenção adequada e, portanto, garantir através de um acompanhamento contínuo uma melhor qualidade de vida às pessoas e diminuir a morbimortalidade causada pelas patologias triadas (MINISTÉRIO DA SAÚDE, 2016).

Segundo o Ministério da Saúde (2016), o PNTN é considerado um programa de grande importância nacional, pois contempla princípios e diretrizes fundamentais do SUS, uma vez, por exemplo, que possui:

- Grande abrangência. Em 2014 atingiu mais de 84% dos nascidos vivos brasileiros na rede pública.
- Ampla implantação. Está implantado em todos os estados brasileiros, coordenados pelas Secretarias de Estado da Saúde e operacionalizado pelas Secretarias Municipais de Saúde.
- Atende aos princípios da universalidade, equidade, integralidade, preservação da autonomia e igualdade da atenção à saúde.
- Acompanhamento integral. Pessoas identificadas com distúrbios são acompanhadas continuamente por especialistas, visando sua saúde integral, redução da morbimortalidade e melhoria da qualidade de vida.

Dessa forma, nas últimas duas décadas o Brasil tem reduzido notavelmente os índices de mortalidade infantil (menores que 5 anos de idade). Em 1990, para cada 1000 nascidos, existiam 62 óbitos, enquanto que em 2012, reduziu-se para 14 óbitos (redução de 77%). De acordo com o Fundo das Nações Unidas para Infância (UNICEF), foi uma das maiores reduções de mortalidade infantil do mundo (MINISTÉRIO DA SAÚDE, 2016).

Em 5 de Junho de 2013 o HU-UFS ampliou os serviços prestados a toda rede pública do estado passando a ser habilitado como Serviço de Referência em Triagem Neonatal (SRTN) em Sergipe através da Portaria Nº 1.082, que atua na fase de detecção de fibrose cística do PNTN (ALMEIDA, 2014).

6.2 O processo de Triagem Neonatal no HU

Como forma de atender aos objetivos e metas propostas pelo PNTN, O HU-UFS estabeleceu e descreveu todas as atividades envolvidas na execução do Programa de Triagem Neonatal. Sendo assim, Almeida (2014) realizou um levantamento de requisitos para identificação dessas atividades, o qual foi utilizado para embasar o modelo de processo inicial mapeado neste trabalho com a notação B4S.ex.

O processo da Triagem Neonatal do HU possui, ao todo, quatro etapas principais, são elas (ALMEIDA, 2014):

1. A coleta de amostras em papel filtro nas Unidades de Saúde.
2. Realização dos exames de triagem no laboratório do HU.

3. Busca pelos casos suspeitos para confirmação através de novos exames.
4. Busca por pacientes para consultas de orientação, atendimento e acompanhamento médico.

O cartão é o principal documento no processo da triagem, o qual contém quatro amostras de sangue retiradas do recém-nascido nos postos de coletas, além de outras informações cadastrais. O cartão é enviado ao HU para análise no Laboratório de Análises Clínicas. A Figura 30 mostra um exemplo do cartão utilizado.

Figura 30: Cartão para cadastro e realização de exames da Triagem Neonatal.

Serviço de Triagem Neonatal HU - Hospital Universitário

PREENCHER COM LETRA DE FORMA

1ª AMOSTRA
RECONVOCADO
CONTROLE

No caso de RECONVOCADO ou CONTROLE, informar no campo abaixo o número de registro da criança no SRTN

REG. SRTN: _____

NOME COMPLETO DO RN _____

NOME COMPLETO DA MÃE _____

TELEFONE(S) _____

ENDEREÇO _____

Bairro _____ **Município** _____

DATA DE NASCIMENTO ____/____/____ **DATA DA COLETA** ____/____/____

SEXO ☐ MASC. ☐ FEM. **GEMELAR** 1ª ☐ 2ª ☐ 3ª ☐ NÃO

ANTIBIÓTICO ☐ SIM ☐ NÃO **PREMATURO** ☐ SIM ☐ NÃO ☐ NÃO SABE **TRANSFUSÃO** ☐ SIM ☐ NÃO ☐ NÃO SABE

POSTO DE COLETA _____ **CIDADE** _____ **TELEFONE** _____

NOME DO COLETOR _____ **ASSINATURA DO RESPONSÁVEL PELO RN** _____

RESULTADOS:

PKU: _____ mg/dl **TSH:** _____ mU/ml

HEMOGLOBINAS: _____ g/dl **BT:** ☐ SIM ☐ NÃO _____ g/dl

NOME / Nº DO POSTO DE COLETA: _____

TELEFONE: _____

NOME DO RECÉM-NASCIDO: _____

DATA DE NASC. ____/____/____ **DATA DA COLETA:** ____/____/____

PREVISÃO DO RESULTADO: _____

Destaque este protocolo e guarde até o recebimento do resultado. Para informações adicionais, Fone: (79) 2105-1752

Fonte: Almeida (2014).

De acordo com Almeida (2014), o processo é lento, devido a morosidade da realização e disponibilização dos resultados, além de ter que atender a diversas normas técnicas e portarias emitidas pelo Ministério da Saúde. O tempo médio de coleta da amostra na Unidade de Saúde e a divulgação dos resultados é de 11 dias (sem levar em consideração o tempo em casos que é necessário a reconvocação do recém-nascido).

Dessa forma, um dos principais desafios é reduzir o período entre a realização dos exames e apresentação do seu resultado, pois quanto mais rápido for identificada a necessidade de tratamento, melhores serão as possibilidades de sucesso. Consequentemente, percebe-se a necessidade de investimentos em tecnologias que suportem a automação desse processo, agilizando suas atividades, reduzindo custos e gerando indicadores de desempenho que permitam um maior controle e gestão do SRTN.

6.3 Modelo B4S.ex da Triagem Neonatal

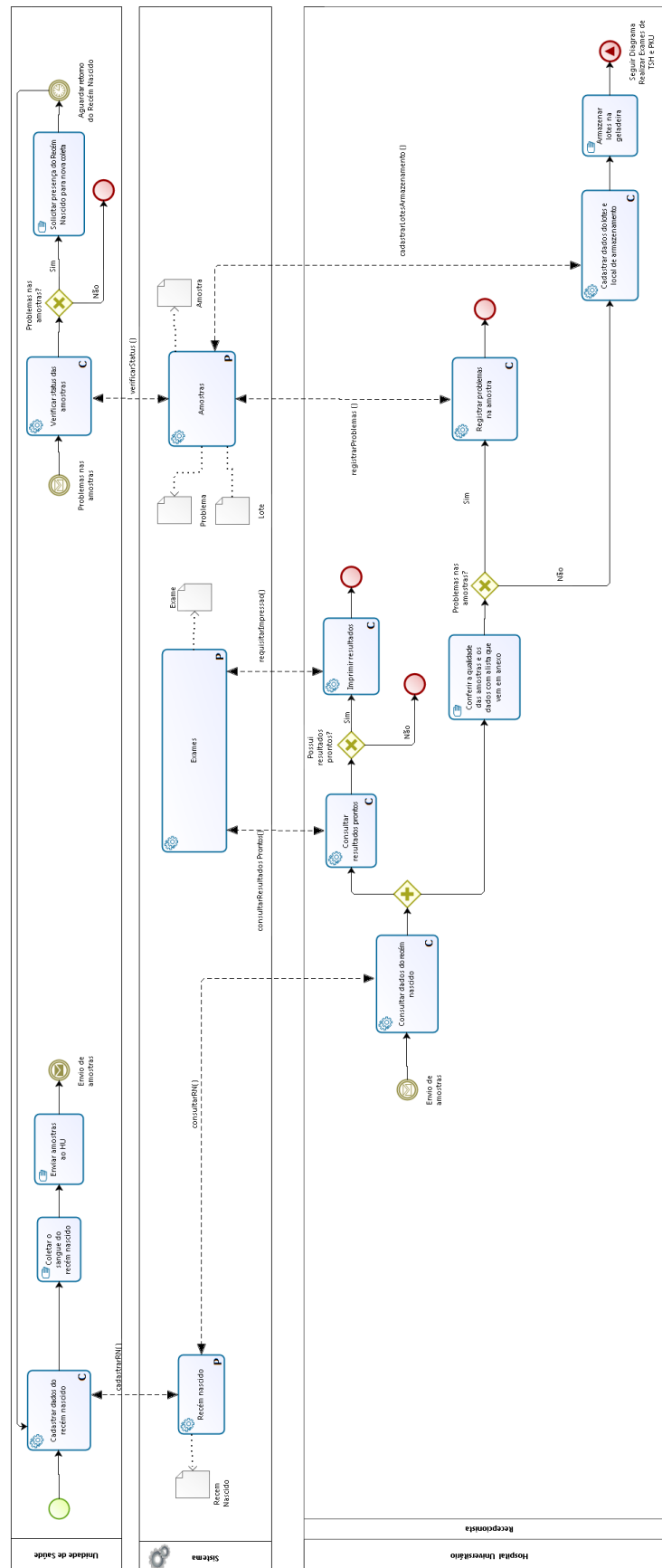
Com base nas atividades apresentadas por Almeida (2014), utilizou-se a extensão proposta neste trabalho, a B4S.ex, para modelagem do processo de negócio da Triagem Neonatal do Hospital Universitário. Como o processo da triagem envolvia diversos setores, internos e externos ao HU, e uma extensa quantidade de atividades, optou-se por modelar apenas a etapa inicial do SRTN, a qual incluiu a coleta na Unidade de Saúde, recepção e análise inicial da qualidade das amostras por parte do hospital. A modelagem do processo é apresentada na Figura 31.

Primeiramente, foram identificados os participantes do processo, foram eles: Unidade de Saúde, responsável pela coleta e contato inicial com o recém-nascido, e o Hospital Universitário que, devido ao escopo reduzido do processo, foi especificado com uma única raia (setor do hospital), a Recepcionista. Para compor a representação do elemento Barramento no contexto da notação B4S.ex, definiu-se o participante Sistema, o qual contém todos os serviços que dão suporte a execução das atividades do processo.

O processo é iniciado na Unidade de Saúde com a tarefa de cadastrar os dados do recém-nascido através da operação “cadastrarRN()” do serviço “Recém nascido”, em seguida a coleta do sangue é realizada e as amostras enviadas ao HU. No hospital, em posse das amostras, a recepção consulta os dados do recém-nascido com a operação “consultarRN()” com o objetivo de verificar a existência de resultados de exames já realizados (operação “consultarResultadosProntos()” do serviço “Exames”). Caso existam exames prontos, a impressão será efetuada pela operação “requisitarImpressão()”.

Paralelamente a consulta de exames já realizados, uma conferência das amostras recebidas é realizada a fim de identificar problemas. Não existindo nenhum problema com as amostras, o cadastro dos lotes e local de armazenamento é realizado através da operação “cadastrarLotesArmazenamento()” do serviço “Amostras”, as amostras são armazenadas e o processo segue para a etapa de execução dos exames (fora do escopo deste exemplo). Caso existam problemas com as amostras, estes problemas são cadastrados utilizando a operação “registrarProblemas()” no serviço “Amostras”. No serviço “Amostra”, a operação “verificarStatus()” é responsável por identificar a presença de um problema cadastrado e indicar que a Unidade de Saúde precisa entrar em contato com os responsáveis do recém-nascido para realizarem uma nova coleta.

Figura 31: Processo de Triagem Neonatal.



Fonte: Autores (2016).

Além da representação do elemento Barramento (representado pelo participante “Sistema”), do provimento e consumo dos serviços (elementos “Consumidor” e “Provedor”) através de suas operações (elemento “Invocação com Retorno”), também estão presentes no modelo B4S.ex as propriedades de cada operação, as quais contém informações como: parâmetros da operação (nomes e tipos de dados), operação conjugada e parâmetros da operação conjugada (nomes e tipos de dados). O Quadro 16 contém o resumo das propriedades de cada operação.

Quadro 16: Propriedades das Operações.

Operação	Parâmetros (nome e tipo)	Operação conjugada	Parâmetros (conjugada) (nome e tipo)
cadastrarRN()	rn (RecemNascido)	obterRN()	confirmacao (boolean)
consultarRN()	idRN (<i>integer</i>)	obterRN()	Rn (RecemNascido)
consultarResultadosProntos()	idRN (<i>integer</i>)	obterResulProntos()	exm (Exame)
requisitarImpressao()	idExame (<i>integer</i>)	obterImpressao()	exm (Exame)
registrarProblemas()	idAmostra(<i>integer</i>) pb (Problema)	confirmacaoPb()	confirmacao (boolean) am (Amostra)
verificarStatus()	idAmostra (<i>integer</i>)	obterStatus()	Am (Amostra)
cadastrarLotesArmazenamento()	lt (Lote)	confirmarCL()	confirmacao (boolean)

Fonte: Autores (2016).

Conforme observado na modelagem e nos parâmetros utilizados nas operações, existem tipos complexos de dados, os quais foram mapeados utilizando uma extensão das propriedades do objeto “*DataObject*” do BPMN, transformando-o no elemento “EntidadeServiço” no B4S.ex (conforme apresentado no capítulo anterior). O Quadro 17 apresenta as entidades modeladas e suas propriedades representando seus atributos, juntamente com os tipos de dados de cada atributo.

Quadro 17: Atributos das Entidades.

Entidade	Atributo	Tipo
RecémNascido	idRN	<i>Integer</i>
	nome	<i>String</i>
	peso	<i>Double</i>
	altura	<i>Double</i>
	dataNascimento	<i>Date</i>
Exame	idExame	<i>Integer</i>
	tipoExame	<i>String</i>
	resultado	<i>String</i>
	rn	RecémNascido
Amostra	idAmostra	<i>Integer</i>
	status	<i>String</i>
	volume	<i>Double</i>
	lote	Lote
Problema	idProblema	<i>Integer</i>
	descricao	<i>String</i>
	am	Amostra
Lote	idLote	<i>Integer</i>
	descricao	<i>String</i>
	local	<i>String</i>

Fonte: Autores (2016).

Conforme visto, o processo de Triagem Neonatal foi mapeado utilizando o metamodelo B4S.ex proposto, compondo o modelo inicial do *framework* BPM4Services. Dessa forma, obteve-se um modelo com a semântica necessária para ser utilizado como entrada nas transformações seguintes do *framework*.

6.4 Aplicação das regras e modelos SoaML gerados

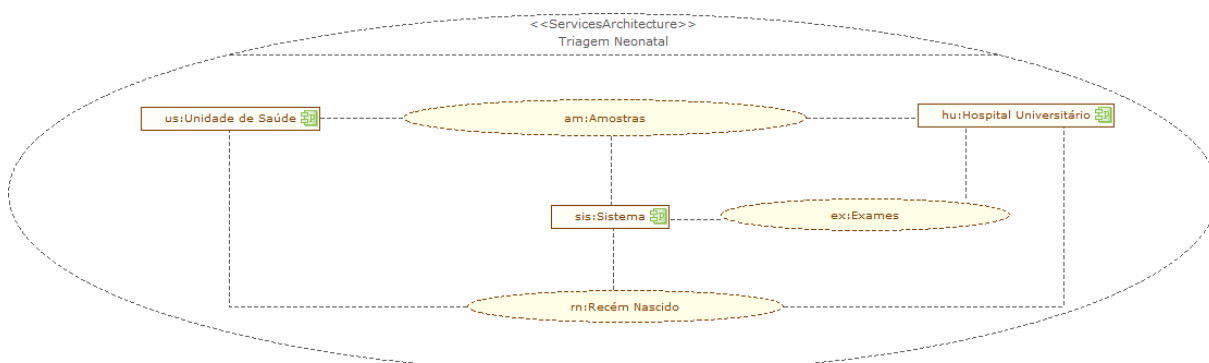
Após o mapeamento do processo de Triagem Neonatal utilizando a extensão proposta neste trabalho, aplicaram-se as regras de mapeamento discutidas no capítulo anterior a fim de obter os diagramas SoaML, são eles: Diagrama de Arquitetura, Diagrama de Participantes, Diagrama de Interfaces e Diagrama de Mensagens. As regras aplicadas serão detalhadas na a seguir.

6.4.1 Diagrama de Arquitetura

O Diagrama de Arquitetura é um dos principais diagramas propostos pela extensão SoaML para representação de um ambiente SOA. Sua principal característica é a apresentação do contexto de interação entre serviços e participantes e como eles se relacionam. Dessa forma, o BPM4Services sugere o Diagrama de Arquitetura como um dos diagramas gerados pelas transformações iniciais a partir do modelo B4S.ex.

Conforme visto na seção anterior, o processo de Triagem Neonatal contém 3 participantes, são eles: Unidade de Saúde, Hospital Universitário e Sistema. A partir da regra RMDA, estes 3 participantes foram convertidos para elementos do tipo *Participant* também no SoaML. Em seguida o participante “Sistema” do tipo Barramento foi identificado e todos os serviços provedores mapeados no modelo B4S.ex foram recuperados, transformados no elemento correspondente a serviços no SoaML e ligados aos participantes que os consomem. No processo de Triagem Neonatal, foram gerados 3 serviços: Amostras, Exames e Recém Nascido. A Figura 32 apresenta o Diagrama de Arquitetura gerado através da aplicação das regras de mapeamento propostas no capítulo anterior.

Figura 32: Diagrama de Arquitetura gerado.



Fonte: Autores (2016).

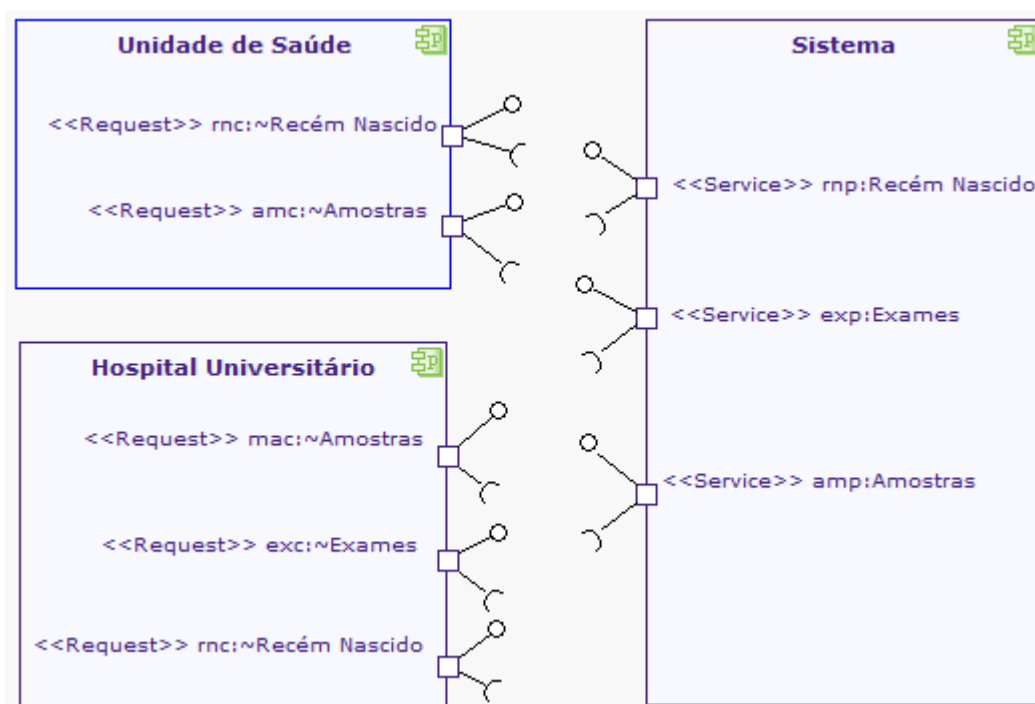
O propósito do diagrama gerado é ilustrar como os tipos de entidade trabalham em conjunto. No exemplo, a Unidade de Saúde trabalha em conjunto com o Hospital Universitário através dos serviços do Recém-Nascido e Amostras (amostras do recém-nascido são enviadas ao HU, tal como visto no processo mapeado, por exemplo), bem como o participante Sistema se correlaciona com todos os demais participantes, provendo todos os serviços da arquitetura.

6.4.2 Diagrama de Participantes

Outro diagrama útil na arquitetura SOA é o de Participantes, o qual representa componentes de software, organizações, sistemas ou pessoas que proveem e consomem serviços através das chamadas “Portas de Serviços”.

No BPM4Services o Diagrama de Participantes é gerado através da regra de mapeamento RMDP. De acordo com a regra, o participante que foi representado pelo elemento do tipo Barramento da extensão B4S.ex é criado sob a perspectiva do provedor de serviços, logo, todos os serviços são criados com o estereótipo “*Service*”. Os participantes Unidade de Saúde e Hospital Universitário, os quais continham os consumidores dos serviços, são representados pelo estereótipo “*Request*”. Para cada “*Service*” ou “*Request*”, existe uma porta no elemento *Participant* correspondente. A Figura 33 ilustra o Diagrama de Participantes gerado através do processo da Triagem Neonatal.

Figura 33: Diagrama de Participantes gerado.



Fonte: Autores (2016).

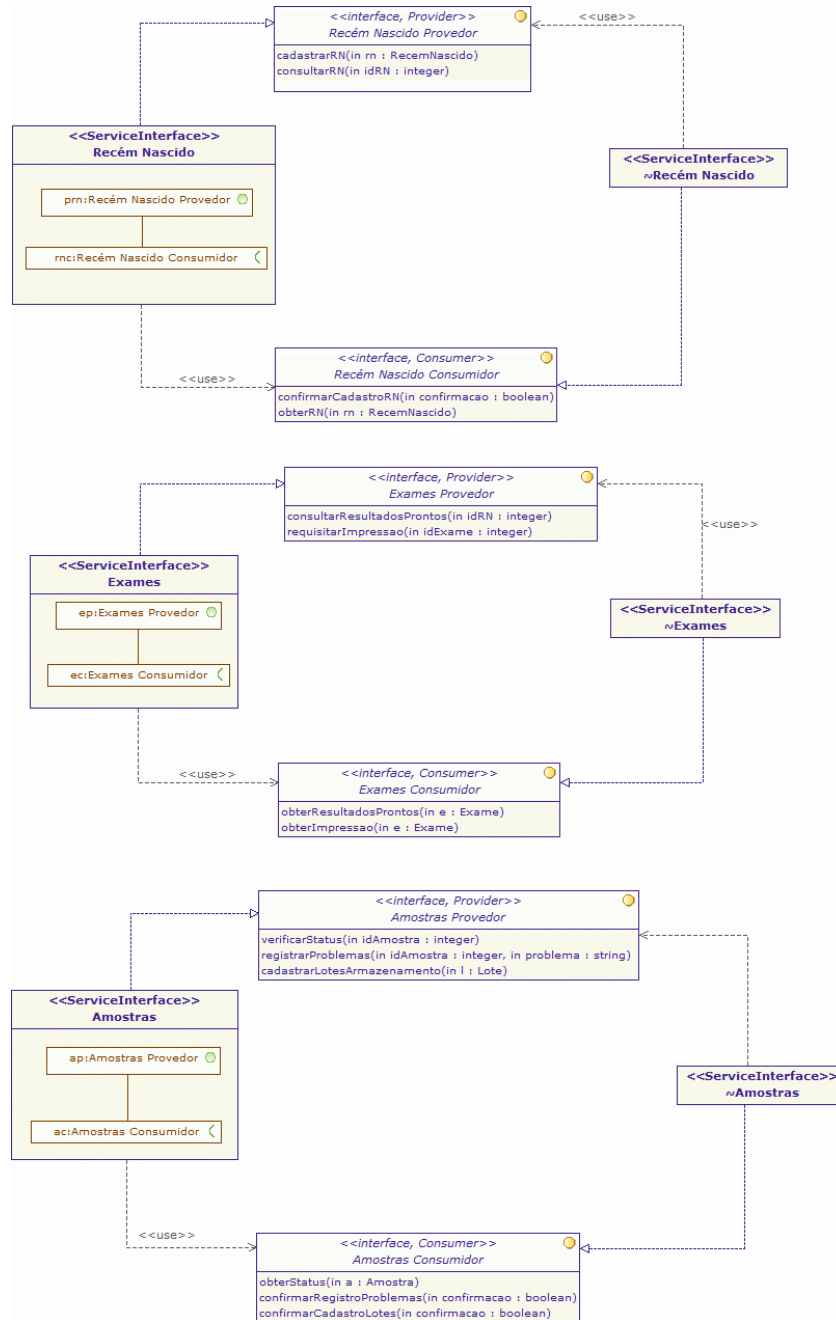
Na Figura 33 é possível identificar, sob a perspectiva de cada participante, quais serviços são consumidos ou providos por cada um. O participante “Sistema” provê os serviços que serão consumidos pelos participantes Unidade de Saúde e Hospital Universitário. Cada serviço contém uma porta e uma interface provedora (representada pelo círculo) e consumidora (representada pelo semicírculo), as quais serão detalhadas no Diagrama de Interfaces.

6.4.3 Diagrama de Interfaces

O diagrama mais complexo desta transformação é o Diagrama de Interfaces, visto que representa a forma como ocorrerá a comunicação entre os consumidores e provedores através das “Services Interfaces”. Para geração automática deste diagrama, informações existentes apenas nas propriedades dos novos elementos da extensão B4S.ex foram essenciais.

A regra RMDI é responsável por fazer a conversão do modelo B4S.ex para o Diagrama de Interfaces. Sendo assim, foram identificados os serviços provedores e os elementos de conexão “Invocação Simples” e “Invocação com Retorno”, pois suas propriedades contêm informações que são utilizadas no diagrama, tais como: operações, parâmetros e operações conjugadas. A Figura 34 apresenta o Diagrama de Interfaces gerado através da regra RMDI.

Figura 34: Diagrama de Interface gerado.



Fonte: Autores (2016).

Uma operação conjugada é utilizada quando se utiliza o elemento “Invocação com Retorno”, já que um serviço que utiliza este componente tem uma comunicação bilateral, ou

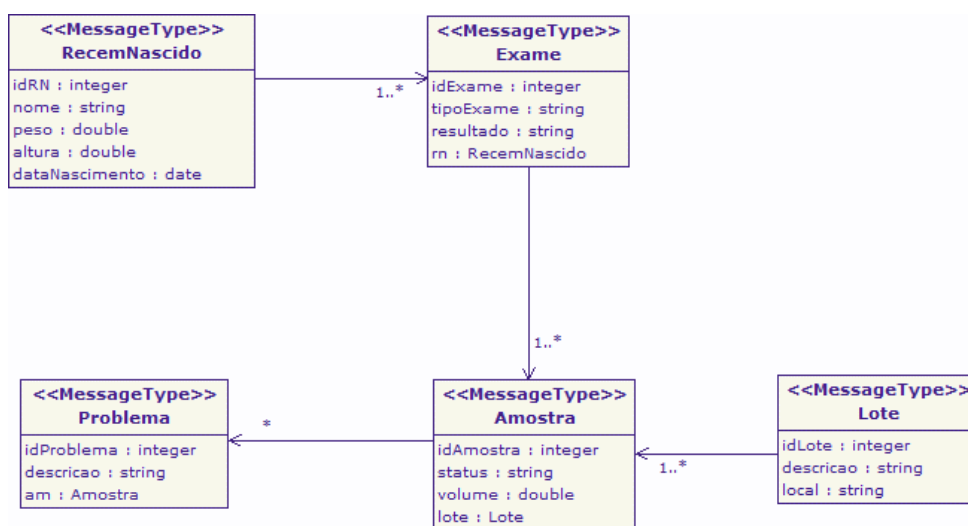
seja, o provedor realiza uma interface provedora e consome uma interface consumidora. A “*ServiceInterface*” do ponto de vista do consumidor é representada no diagrama de forma inversa da perspectiva do provedor: usa a interface provedora e realiza a interface consumidora.

Na imagem são representadas as “*ServicesInterfaces*” de cada serviço mapeado no processo da Triagem Neonatal, destacando as interfaces que são relacionadas na execução deste serviço. Por exemplo, “Exames” é uma “*ServiceInterface*” que contém as interfaces “Exame Provedor” e “Exame consumidor”, cada interface contém as operações disponíveis para invocação. Ou seja, para invocar o serviço “Exames”, o consumidor precisa conhecer a interface “Exame Provedor”, verificando as operações disponíveis e seus parâmetros. Por outro lado, o provedor precisa conhecer como passará os dados para o consumidor através da interface “Exames Consumidor”.

6.4.4 Diagrama de Mensagens

O Diagrama de Mensagens é derivado da regra RMDM, a qual tem como objetivo identificar todos os elementos de dados do processo e representa-los através de uma estrutura de classes. Para tanto, são verificados os elementos do tipo “*DataObject*” estendido, ou “EntidadesServiços” (como determina o metamodelo B4S.ex), a fim de buscar em suas propriedades todos os atributos referentes aquele componente de dados. A Figura 35 apresenta o Diagrama de Mensagens gerado.

Figura 35: Diagrama de Mensagens gerado.



Fonte: Autores (2016).

Após a execução das regras de mapeamento, o Diagrama de Mensagem gerado identificou basicamente 5 elementos “*MessageType*”, os quais representam os dados manipulados pelos serviços, são eles: “RecemNascido”, “Exame”, “Problema”, “Amostra” e

“Lote”. As relações entre os novos elementos são formadas a partir da leitura de seus atributos, caso exista algum tipo pertencente a outra classe, então a relação é estabelecida, por exemplo, a classe “Amostra” contém o atributo “lote” do tipo “Lote”, logo, cria-se a relação entre as classes “Amostra” e “Lote”.

6.5 Análise da aplicação do BPM4Services no caso exemplo

A aplicação do BPM4Services no processo da Triagem Neonatal do HU-UFS, por meio de um caso exemplo, possibilitou realizar uma análise mais aprofundada nas fases iniciais do *framework* proposto. Conforme propõe o BPM4Services, o modelo B4S.ex resultante do mapeamento do processo da Triagem Neonatal foi utilizado como entrada para execução das transformações e geração dos diagramas SoaML com base nas RMDA, RMDP, RMDI e RMDM.

Como não foram desenvolvidas ferramentas que apoiassem o processo de transformação, todas as transformações e gerações dos diagramas SoaML foram feitas manualmente. Contudo, ainda teve como resultado uma análise do *framework*, identificando e avaliando suas características, limitações e contribuições.

Primeiramente, percebeu-se que o modelo gerado pela extensão desenvolvida tem duas perspectivas de análise: uma voltada ao negócio e outra para tecnologia. Ou seja, dois profissionais envolvem-se na criação do modelo B4S.ex, o primeiro é o dono do processo e, geralmente, detém todas as regras de negócio e conhecimento do funcionamento do processo, o segundo é o profissional da área de tecnologia, o qual deve conduzir o dono do processo durante a construção do modelo, adicionando propriedades e elementos que são pertinentes às transformações subsequentes das fases do *framework*.

Dessa forma, a atenção primária dos envolvidos na automação de um processo utilizando o BPM4Services deve ser no modelo B4S.ex, de forma que todos os elementos utilizados em sua representação estejam em conformidade semântica com o seu metamodelo. Sendo assim, percebeu-se que existe a necessidade de entender os novos componentes da extensão B4S.ex, como por exemplo, suas propriedades, restrições e formas de utilizar. Contudo, acredita-se que a curva de aprendizagem do B4S.ex seja menor do que a das notações mais técnicas como a UML/SoaML, pois o B4S.ex descende do BPMN, o qual é um padrão amplamente difundido no mercado, seja entre os profissionais de TI ou não. Dessa forma, o conhecimento de UML/SoaML são necessários apenas nas definições das regras de mapeamento dentro do *framework* BPM4Services no momento de sua implementação.

O caso exemplo também contribuiu para evolução do framework. Durante sua condução alguns componentes do B4S.ex foram ajustados, bem como algumas regras de mapeamento para geração dos diagramas SoaML. Por exemplo, percebeu-se a necessidade de especializar o componente Invocação, uma vez que existem consumidores que podem ou não ter um retorno por parte do provedor, consequentemente, realizaram-se ajustes nas regras de mapeamento para geração do diagrama de interfaces (RMDI).

Por fim, o escopo do BPM4Services é muito mais amplo do que este caso exemplo conseguiu abordar. Caso o *framework* fosse completamente implementado e utilizado no processo da Triagem Neonatal do HU-UFS, *web services* seriam criados a partir dos diagramas SoaML (via transformação ATL). Em seguida, os *web services* seriam invocados através dos *scripts* BPEL e validados por meio dos casos de testes gerados. Todo este processo aconteceria com a menor interação humana possível, exceto pela etapa do desenvolvimento do modelo B4S.ex, abordada neste caso exemplo. Dessa forma o BPM4Services, não somente aumentaria a velocidade de produção de uma automação para o processo de negócio, como também auxiliaria sua adaptação frente a mudanças, visto que bastaria alterar o modelo B4S.ex e realizar as transformações necessárias para que toda a regra de negócio fosse alterada.

6.6 Considerações Finais do Capítulo

Este capítulo apresentou um caso exemplo utilizado para ilustrar uma das fases do *framework* BPM4Services. O objeto de estudo deste caso exemplo foi o Programa Nacional de Triagem Neonatal do Hospital Universitário da Universidade Federal de Sergipe. Para entender o PNTN utilizou-se o trabalho de Almeida (2014), o qual descreve os requisitos necessários para modelagem do processo da Triagem Neonatal do HU-UFS.

Dessa forma, iniciou-se o caso exemplo com a modelagem do processo de negócio referente ao SRTN utilizando a extensão do BPMN proposta neste trabalho (B4S.ex) e, em seguida, através da aplicação das Regras de Mapeamento desenvolvidas no capítulo anterior, gerou-se os diagramas SoaML de Arquitetura, Participantes, Interfaces e Mensagens.

O final do capítulo apresentou algumas considerações sobre a utilização do BPM4Services, destacando algumas contribuições de sua aplicação para o HU-UFS, bem como os benefícios do caso exemplo para aprimoramento do *framework* proposto. No próximo capítulo serão apresentadas as considerações finais deste trabalho, principais contribuições, dificuldades, limitações da pesquisa e trabalhos futuros.

CAPÍTULO 7

CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

O aumento da competitividade entre organizações tem gerado uma preocupação constante em aprimorar processos continuamente. Os principais objetivos organizacionais se caracterizam pela execução de atividades que agregam valor à própria empresa junto aos seus clientes. Nesse contexto, várias técnicas e abordagens podem ser utilizadas, dentre elas o BPM. O que diferencia BPM das demais abordagens de gestão é sua capacidade de obter controle e visibilidade do processo ponta a ponta sobre suas necessidades atuais e futuras, permitindo agilidade e comunicação para inovação dos processos através do uso de tecnologias da informação (CAPOTE, 2011).

Entretanto, ainda existem dificuldades para lidar com ambientes dinâmicos, os quais exigem uma constante evolução dos modelos de processos de negócio. Uma das dificuldades é a distância entre o domínio do problema e o domínio da implementação de uma solução, aumentando a dificuldade no desenvolvimento de softwares complexos e, conseqüentemente, o número de problemas gerados durante sua implementação.

Com o objetivo de diminuir a distância entre domínio do problema e domínio da solução, este trabalho propôs o BPM4Services, um *framework* dirigido a modelos para automação de processos de negócio baseado em uma arquitetura orientada a serviços. Com isso, buscou-se construir uma abordagem mais eficiente e sistematizada, com melhores mecanismos de controle e maior produtividade para desenvolver soluções integradas aos conceitos de BPM, SOA e MDE.

O BPM4Services foi delineado com base na arquitetura MDA da OMG, dessa forma, os modelos utilizados no *framework* estão distribuídos entre os níveis de abstração CIM, PIM, PSM e código. Seu principal objetivo é possibilitar a geração de uma solução concreta baseada em SOA a partir de transformações dos modelos de processo de negócio. Para tanto, foi necessário definir uma extensão da notação BPMN para atender às necessidades do *framework*, denominada B4S.ex.

A extensão B4S.ex foi criada em conformidade com o metamodelo BPMN V2.0, o qual possibilita a criação de extensões de seus elementos e suas propriedades. Também foi utilizada uma metodologia formal para sua criação, proposta por Stroppi, Chiotti e Villarreal (2011) e complementada por Braun *et al.* (2014). Em seguida, foram criadas as regras de mapeamento

entre o metamodelo B4S.ex e o metamodelo SoaML. Sendo assim, obteve-se, a partir do modelo de processo de negócio B4S.ex, os diagramas SoaML de Arquitetura, Participantes, Interfaces e Mensagens.

Para demonstrar a utilização do BPM4Services, utilizou-se o caso exemplo do processo de Triagem Neonatal do Hospital Universitário da UFS. O processo foi mapeado com a extensão B4S.ex e, em seguida, foram aplicadas as regras de mapeamento definidas para geração dos diagramas SoaML.

O caso exemplo demonstrou que é possível utilizar a extensão proposta para mapear todo um processo de negócio, garantindo sua integridade e conformidade com os conceitos e premissas do padrão BPMN. Também foi possível observar que as regras de mapeamento cobriram grande parte dos componentes utilizados nos diagramas SoaML escolhidos e mantiveram suas semânticas. Por fim, permitiu detectar as limitações e dificuldades encontradas neste trabalho e em possíveis trabalhos futuros, as quais serão apresentados nas seções seguintes.

7.1 Principais Contribuições

Dentre as contribuições deste trabalho, pode-se destacar a identificação do estado da arte na Gestão de Processos com foco na automação (capítulo 2), o qual contém, dentre outras informações, uma análise histórica, apresentando um retrospecto da evolução dos principais conceitos que contribuíram para o surgimento da definição formal da Gestão de Processos de Negócio. Da mesma forma, o capítulo 3 apresenta uma visão geral das abordagens dirigidas a modelos e suas principais definições.

Outra contribuição é a análise das abordagens para automação de processo de negócio sob a perspectiva da Engenharia Dirigida a Modelos, no capítulo 4, através de uma Revisão Sistemática de Literatura. Foram identificadas propostas de trabalhos que apresentassem alguma forma de integrar os conceitos de BPM, SOA e MDE através de *frameworks*, ferramentas, metodologias, dentre outros. Sendo assim, destacaram-se as principais dificuldades e limitações existentes nas abordagens dirigidas a modelos para automação de processo de negócio.

A principal contribuição deste trabalho é a proposta do BPM4Services, o qual tem por objetivo atender às principais dificuldades encontradas e integrar BPM, SOA e MDE para compor uma possível solução para automação de processos. Na seção 5.2, como parte do desenvolvimento do BPM4Services, realizou-se um estudo para elencar as principais extensões

BPMN e suas características, como por exemplo a metodologia utilizada para sua criação, seu escopo ou área de atuação, sua relação com abordagens dirigidas a modelos, dentre outras. Como resultado das necessidades identificadas no estudo, surgiu o B4S.ex, extensão BPMN incorporada ao BPM4Services para facilitar as transformações entre modelos no *framework*.

Este trabalho também contribuiu com uma análise entre os metamodelos B4S.ex (metamodelo estendido BPMN) e SoaML, apresentando um mapeamento entre eles a fim de gerar diferentes diagramas SoaML a partir de um modelo B4S.ex.

No capítulo 6, houve uma demonstração de parte do *framework* BPM4Services, através de um caso exemplo, da modelagem do processo com a extensão B4S.ex proposta e a aplicação das regras de transformações para geração dos modelos SoaML.

7.2 Limitações da Pesquisa

Uma limitação deste trabalho refere-se à aplicação do BPM4Services em apenas um caso exemplo. Para uma validação mais expressiva do *framework* seriam necessárias aplicações em estudos de casos em diferentes cenários e organizações reais, os quais determinariam formalmente aspectos a serem avaliados na automação de seus processos de negócio com a utilização do BPM4Services. Com as informações geradas nos estudos de casos seria possível analisá-las e, portanto, determinar a viabilidade prática do *framework* na automação de processos de negócio.

Outra dificuldade encontrada foi a ausência de uma ferramenta que implementasse o *framework* BPM4Services e o metamodelo da extensão B4S.ex, pois demandaria tempo e esforço para implementar uma ferramenta com tantas características e peculiaridades. Dessa forma, conseqüentemente, a construção do modelo inicial B4S.ex e execução de suas transformações e geração dos diagramas SoaML durante o caso exemplo foram realizadas manualmente.

Por fim, o BPM4Services apresenta uma possível solução para todo o ciclo de vida da automação de um processo de negócio, contudo, apenas as fases iniciais do *framework* são detalhadas neste trabalho. O processo de criação do metamodelo utilizado na camada CIM do BPM4Services foi a fase do trabalho mais aprofundada, seguida pela fase de definição das regras de mapeamento para compor as transformações do modelo B4S.ex em diagramas SoaML. As demais fases do BPM4Services não foram apresentadas com detalhes suficientes para uma implementação imediata, sendo necessárias algumas informações adicionais.

7.3 Trabalhos Futuros

Como forma de ultrapassar algumas das limitações vistas anteriormente, foram identificados possíveis trabalhos futuros, são eles:

- Criação das regras de mapeamento entre os modelos das demais transformações do BPM4Services.
- Detalhamento da geração dos testes automatizados propostos no *framework*.
- Geração de novos diagramas SoaML através da extensão B4S.ex.
- Realização de um estudo de caso para validação formal do BPM4Services.
- Desenvolvimento de uma ferramenta de modelagem que atenda aos requisitos da extensão B4S.ex.
- Desenvolvimento de uma ferramenta para dar suporte a todas as etapas do BPM4Services: definição de regras de mapeamento, transformações e testes automatizados.
- Disponibilização dessa ferramenta para a comunidade acadêmica através de ambientes amplamente conhecidos, como o Portal do *Software* Público, de forma a favorecer a disseminação da abordagem utilizada, bem como discussões e melhorias do BPM4Services por outros pesquisadores
- Publicações dos resultados dos estudos em conferências e periódicos a fim de compartilhar conhecimento, bem como buscar discussões e sugestões de melhorias na comunidade acadêmica.

REFERÊNCIAS

- ABPMP BPM CBOK v3.0. *Guide to the Business Process Management Common Body of Knowledge V3.0*. ABPMP Internacional, 2013.
- ALMEIDA, C. C. J. *Qualitas: Um Modelo de Processo de Desenvolvimento Orientado a Modelos*. 2014.
- AMELLER, D. *Non-Functional Requirements as drivers of Software Architecture Design*. 2009.
- AMSDEN, J. *Modeling With SoaML, the Service-Oriented Architecture Modeling Language: Part 1, service identification*, 2010. Disponível em <<http://www.ibm.com/developerworks/rational/library/09/modelingwithsoaml-1/>>. Último acesso em 28 de dezembro de 2015.
- AMSTEL *et al.* *Performance in Model Transformations: Experiments with ATL and QVT*. Springer Berlin Heidelberg, 4th International Conference, Zurich, 2011.
- ATL. ATL – A Model Transformation Technology. Disponível em <<http://www.eclipse.org/atl>>. Último acesso em 05 de janeiro de 2016.
- BALDAM, R. *et al.* *Gerenciamento de Processos de Negócio BPM – Business Process Management*. Editora Érica, 2009.
- BARROS, A. J. S. e LEHFELD, N. A. S. *Fundamentos de Metodologia: Um Guia para a Iniciação Científica*. 2 Ed. São Paulo: Makron Books, 2000.
- BARROS, R. S. C. *Definição de uma Framework para Municípios de Pequena Dimensão*. Universidade do Porto, dissertação, 2009.
- BÉZIVIN, J. *Model Driven Engineering: As Engineering Technical Space*. In: *Generative and Transformational Techniques in Software Engineering*, 2006.
- BOEHM, B. A view of 20th and 21st century software engineering. *Proceeding of the 28th international conference on Software engineering - ICSE '06*, p. 12, 2006.
- BOSEMS, S. *A Performance Analysis of Model Transformations and Tools*. University of Twente, Holanda, 2011.
- BRAUN, R. *et al.* *BPMN4CP: Design and Implementation of a BPMN Extension for Clinical Pathways*. IEEE International Conference on Bioinformatics and Biomedicine, 2014.
- CAPOTE, G. *Guia para Formação de Analistas de Processos*. 1ª Edição, 2011.
- CASANAVE, C. *Service Oriented Architecture Using the OMG SoaML Standard. Model Driven Solutions*, 2009. Disponível em <

<http://www.omg.org/news/whitepapers/EnterpriseSoaML.pdf> >. Último acesso em 28 de dezembro de 2015.

CHANG, J. F. *Business Process Management Systems: Strategy and Implementation*. Auerbach Publications Taylor & Francis Group, 2006.

CZARNECKI, K.; HELSEN, S. *Feature-Based of Model Transformation Approaches*. University of Waterloo, Canada, 2006.

DAHMAN, K.; CHAROY, F.; GODART, C. *Generation of Component Based Architecture from Business Processes: Model Driven Engineering for SOA*. 2010 Eighth IEEE European Conference on Web Services, p. 155–162, 2010.

_____. *Towards Consistency Management for a Business-Driven Development of SOA*. 2011 IEEE 15th International Enterprise Distributed Object Computing Conference, p. 267–275, 2011.

_____. *Alignment and Change Propagation between Business Processes and Service-Oriented Architectures*. 2013 IEEE International Conference on Services Computing, p. 168–175, 2013.

DAVENPORT, T. H. *Process Innovation*. Harvard Business School Press, Boston, MA, 1993.

DAVENPORT, T. H.; SHORT, J. E. *The New Industrial Engineering: Information Technology and Business Process Redesign*. Sloan Management Review, 1990).

DAVENPORT, T. H.; STODDARD, D. B. *Reengineering: Business Change of Mythic Proportions?* MIS Quarterly, 1994.

DELGADO, A.; GUZMÁN, I. G. DE; PIATTINI, M. *From BPMN business process models to SoaML service models: a transformation-driven approach*. 2nd International Conference on Software Technology and Engineering (ICSTE), p. 314–319, 2010.

DEMIR, A. *A Comparison of model-driven architecture and software factories in the context of model-driven development*. Proc. - Joint Meeting of the 4th Workshop on Model-Based Dev. of Computer-Based Systems and the 3rd Int. Workshop on Model-Based Methodologies for Pervasive and Embedded Software, MBD/MOMPES 2006, p. 75–83, 2006.

DRAFFIN, A., *Methodology vs framework – why waterfall and agile are not methodologies*. 2007. Disponível em <<https://goo.gl/gHNw8W>>. Último acesso em 28 de dezembro de 2015.

FONSECA, F. L.; OLIVEIRA T. C.; PEREIRA E. B. *Uma Extensão para modelagem de Processos de Desenvolvimento de Software: BPMNt*. VII Simpósio Brasileiro de Sistemas de Informação, 2011.

FRANCE, R.; RUMPE, B. *Model-Driven Development of Complex Software: A Research Roadmap*. Workshop on the Future of Software Engineering (FOSE 2007), at the 29th

International Conference on Software Engineering (ICSE 2007), Minneapolis, Minnesota, USA, p. 37–54, 2007.

FRIEDENSTAB, J. P. *et al.* *Extending BPMN for Business Activity Monitoring*. 45th Hawaii International Conference on System Sciences, 2012.

FUGITA, H. S. e HIRAMA, K. *SOA modelagem, análise e design*. Elsevier Editora, Rio de Janeiro, 2012.

GAO F.; DERGUECH, W.; ZAREMBA, M. *Extending BPMN 2.0 to Enable Links between Process Models and ARIS Views Modeled with Linked Data*. Springer-Verlag Berlin Heidelberg, 2011.

GOLDNER, S.; PAPPROTH A. *Extending the BPMN Syntax for Requeriments*, Springer-Verlag Berlin Heidelberg, 2011.

GONÇALVES, J. E. L. As empresas são grandes coleções de processos. ERA – Revista de Administração de Empresas v.40, n.1 p. 6-19, São Paulo, 2000.

GONÇALVES, J. E. L. DREYFUSS, C. *Reengenharia das empresas: passando a limpo*. São Paulo, Atlas, 1995.

GRAHAM, M. LEBARON, M. *The Horizontal Revolution*. San Francisco: Jossey-Bass, 1994.

HARMON, P. *The Scope Evolution of Business Process Management*. In *Handbook on Business Process Management 1*, Springer, Berlin, 2010.

MANSIR, B. E.; SCHACHT, N. R. *Total Quality Management: a guide to implementation*. Logistics Management Institute, Pennsylvania State University, 1989.

MUSTAFA, N. M. F.; BOCHMANN, G. V. *Transforming dynamic behavior specifications from activity diagrams to BPEL*. *Proceedings of 2011 IEEE 6th International Symposium on Service Oriented System (SOSE)*, n. Sose, p. 305–311, 2011.

HAMMER, M. *Reengineering work: don't automate, obliterate*. *Harvard Business Review*, 1990.

_____. *Process Management and the Future of Six Sigma*. *MIT Sloan Management Review*, 2002.

HAMMER, M.; CHAMPY, J. *Reengineering the Corporation*. New York: HarperBusiness, 1994.

HARMON, P.; WOLF, C. *The State of Business Process Management*. BPTrends, 2014. Disponível em: <<http://www.bptrends.com/bpt/wp-content/uploads/BPTrends-State-of-BPM-Survey-Report.pdf>>. Último acesso em 10 de dezembro de 2015.

HURWITZA *et al.* Arquitetura Orientada a Serviços – SOA para leigos. Editora ALTA BOOKS. 2009.

HUTCHINSON, J.; WHITTLE, J.; ROUCENFIELD, M. *Model-driven engineering practices in industry: Social, organizational and managerial factors that lead success or failure. Science of Computer Programming*, 2014.

HU-UFS. Hospital Universitário da Universidade Federal de Sergipe, 2012. Disponível em: <<http://www.ebserh.gov.br/web/hu-ufs>>. Acesso em 16/10/2016.

INAZAWA, Rafael. A Aplicação do BPM para Automação de Processos de Negócio nas Organizações – Estudo de Caso: Projeto NEW_RCMS. São Paulo, 2009.

JOHNSON, R. E. (1991). *Reusing Object-Oriented Design*, University of Illinois, Technical Report UIUCDCS 91-1696, 1991.

JOSUTTIS, N. M. SOA na Prática. A Arte da Modelagem de Sistemas Distribuídos. Editora ALTA BOOKS, 2008.

KAMOUN, F. *A Roadmap Towards the Convergence of Business Process Management and Service Oriented Architecture*. Ubiquity Journal, v. 8, n. 14, p. 1-8. 2007.

KITCHENHAM, B. *Procedures for Performing Systematic Reviews*. Software Engineering Group, Keele University. 2004.

KLEPPE, A.; WARMER, J.; BAST, W. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison Wesley Professional, 2003.

FRANCESQUINI, E.; KON, F. Orquestração e Coreografias de Serviços Web. Congresso Brasileiro de Software, Salvador, 2010.

LUSK, S.; PALEY, S.; SPANYI, A. *The Evolution of Business Process Management as a Professional Discipline*. ABPMP, 2005. Disponível em: <<http://www.bptrends.com/publicationfiles/06-05%20WP%20ABPMP%20Activities%20-%20Lusk%20et%20al2.pdf>>. Último acesso em: 11 de dezembro de 2015.

MARZULLO, F. P. SOA na prática: inovando seu negócio por meio de soluções orientadas a serviços. Novatec Editora, São Paulo, 2009.

MEDEIROS, E. R. NovaStudio: Gerador de código usando a Arquitetura Dirigida pelos Modelos (MDA). Universidade do Rio Grande do Sul, 2009.

MELLOR *et al.* *MDA Distilled: Principles of Model-Driven Architecture*. Addison Wesley, 2004.

MILLER, J.; MUKERJI, J. *MDA Guide Version 1.0.1*. OMG, 2003.

MILICEV, D. *Model-Driven Development with Executable UML*. Published by Wiley Publishing, Inc., Indianapolis, Indiana, 2009.

MINISTÉRIO DA SAÚDE. Triagem Neonatal Biológica, Manual Técnico. Brasília/DF. 2016.

MIRANDA, P. A. P. SOA – Arquitetura Orientada a Serviços. Universidade da Amazônia – UNAMA. Belém, 2008.

OMG. Object Management Group. *Model Driven Architecture (MDA)*. n. June, p. 1–15, 2003.

_____. *Service oriented architecture Modeling Language (SoaML) – Specification for the UML Profile and Metamodel for Services (UPMS)*, 2009.

_____. *Business Process Model and Notation (BPMN) v.2.0*, 2011. Disponível em: <http://www.omg.org/spec/BPMN/2.0/PDF/>. Último acesso em: 29 de dezembro de 2015.

OASIS. *Web Services Business Process Execution Language Version 2.0*. Oasis Standard, 2007. Disponível em: <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>. Último acesso em: 05 de janeiro de 2016.

PELTZ, C. *Web Service Orchestration and Choreography*. WSJ Feature, 2003.

RAMALHO, F. S. Curso de Desenvolvimento Dirigido a Modelos, 2008. Disponível em <<http://www.dsc.ufcg.edu.br/~franklin/disciplinas/2008-1/DDM/index.php/Main/HomePage>>. Último acesso em 24 de janeiro de 2016.

SAEEDI, K.; ZHAO, L.; SAMPAIO, P, R, F. *Extending BPMN for Supporting Customer-Facing Service Quality Requirements*. IEEE Internacional Conference on Web Services, 2010.

SANDERS, N. R.; REID, R. D. *Operations Management: An Integrated Approach*, 5th Edition, John Wiley & Sons, 2012.

SANTOS, W. C. Uma Abordagem Dirigida por Modelos para Gerência de Variabilidades e Execução de Processos de Software. Dissertação (Mestrado) - Universidade Federal do Rio Grande do Norte, 2011.

SELIC, B. *The Pragmatics of Model-Driven Development*. Published by IEEE Computer Society IEEE Software, 2003.

SHEN, Y.; LAU, L. K. *Total Quality Management VS. Business Reengineering*. Virginia Polytechnic and State University, 1995. Disponível em: <http://www.pacis-net.org/file/1995/45.pdf>. Último acesso em: 11 de dezembro de 2015.

SILVA, E. L. MENEZES, E. M. Metodologia da Pesquisa e Elaboração de Dissertação. Universidade Federal de Santa Catarina, 2001.

SILVA, L. M. Aplicando a Composição e Orquestração de Serviços na Organização de Sistemas. Centro Federal de Educação Tecnológica do Rio Grande do Norte, 2007.

SILVER, B. *BPMN Method and Style*. Cody-Cassidy Press, 2009.

SIQUEIRA, F L. Transformação de um modelo de empresa em um modelo de casos de uso seguindo os conceitos da engenharia dirigida por modelos. Tese de doutorado. Escola Politécnica da Universidade de São Paulo, São Paulo, 2011.

SMITH, H.; FINGAR, P. *Business Process Management (BPM): The Third Wave*, Meghan-Kiffer Press; 1st Edition, 2007.

SINDHGATTA, R. *Interleaving Execution into Model Driven Service Design*. 2013 IEEE 20th International Conference on Web Services, p. 364–371, 2013.

SOSA-SANCHEZ, E. et al. *Service Discovery Using a Semantic Algorithm in a SOA Modernization Process from Legacy Web Applications*. 2014 IEEE World Congress on Services, n. i, p. 470–477, 2014.

SOUZA, A; RABELO, R. J. *An Approach for a More Agile BPM-SOA Integration Supported by Dynamic Services Discovery*. International Enterprise Distributed Object Computing Conference Workshops. 2010.

STAAB, S. et al. *Model Driven Engineering with Ontology Technologies*. University of Koblenz-Landau, Germany, 2010.

STAHL et al. *Model-Driven Software Development: Technology, Engineering, Management*. John Wiley & Sons, 2006.

STROPPI, L. J. R.; CHIOTTI, O.; VILLARREAL P. D. *Extending BPMN 2.0 and Tool Support*. Springer-Verlag Berlin Heidelberg, 2011.

SUPULNIECE, I.; BUSINSKA, L.; KIRIKOVA, M. *Towards Extending with the Knowledge Dimension*. Springer-Verlag Berlin Heidelberg, 2010.

TOUZI, J.; BÉNABEN, F.; PINGAUD, H. *Model transformation of collaborative business process into mediation information system*. IFAC World Congress, p. 13857–13862, 2008.

VENTURA, M. M. O Estudo de Caso como Modalidade de Pesquisa. *Pedagogia Médica*. 2007.

VIDOTTI, E.; SANTOS, P.; VIDOTTI, S. *Reengenharia, Qualidade Total e Unidades de Informação*. Londrina, v.3, n.1, p. 51-54. 1998.

VIEIRA, A. S. Identificação de Diretrizes para Construção de Meta-modelos na Infra-estrutura de MDA. Universidade Federal de Campina Grande, 2010.

VOS, G. *Issues of Iterative MDA-Based Software Development Process*. Dissertação (Mestrado) - *University of Twente*, 2008.

WAZLAWICK, R. S. *Metodologia de Pesquisa para Ciência da Computação*. Rio de Janeiro, RJ, Editora: Elsevier. 2008.

WESKE, M. *Business Process Management: Concepts, Languages, Architectures*. Springer-Verlag Berlin Heidelberg, 2007.

WOODSIDE, M.; PETRIU, D.; FAISAL, A.; A Systematic Approach for Composing General Middleware Completions to Performance Models in Computer Performance Engineering. Proc. European Performance Engineering Workshop EPEW14, Florence. LNCS vol. 8721, Springer, p. 33-44, 2014.